COMPUTERS CHALLENGE AMERICA'S CUP
MICRO DIET: BETTER HEALTH THROUGH ELECTRONICS
COMPUTER MODELS IN PSYCHOLOGY
A BEGINNER'S GUIDE TO PERIPHERALS

ROM
COMPUTER APPLICATIONS FOR LIVING

STAR WARS
Centerfold
R2-D2

Volume I, Number 6
December 1977/$2.00

LSI-11
Nan Ardolino

$995
$175
$325
$395
SWTPC MINIFLOPPY DISK SYSTEM
SWTPC CT-64 TERMINAL SYSTEM
SWTPC 6800 COMPUTER SYSTEM
ROM
BYTE
kilobaud
INTERFACE
Personal Computing
Microcomputer INTERFACE
COMPUTER Games & Quizzes
SCIENTIFIC AMERICAN
creative computing

Your computer system needn't cost a fortune.

Some computer kits include little more than a mother board and a front panel, and you pay extra for everything else you need to make an operating computer.

SWTPC doesn't do it that way, so you can get your Southwest Technical 6800 Computer up and running at a bargain cost compared with most other systems. It comes complete at $395 with features that cost you extra with many other systems.

The Extras You Get

These extras include 4K of random-access memory, a mini-operating system in read-only memory, and a serial control interface. They give you 1) a considerable amount of working memory for your programs, 2) capability through the mini-operating system to simply turn on power and enter programs without having to first load in a bootstrap loader, and 3) an interface for connecting a terminal and beginning to talk with your computer immediately.

Low-Cost Add-Ons

Now that you have a working computer, you'll probably want to add at least two features soon, more memory and interfaces for needed accessory equipment. Memory for our 6800 is another bargain. You can get 4K memory boards for just $100 and 8K boards for only $250.

Our interfaces cost little compared with many other systems. For just $35 you can add either a serial or parallel interface board. (And you won't have to buy several interfaces on a costly board to get just the one you want.)

Peripheral Bargains

Your computer is no good without at least a terminal for entering data and viewing computer output, and you will probably want a good method of storing programs and data.

We offer you a line of high-quality peripherals at low prices. (You can prove this by just comparing prices.)

Buy our CT-64 Video Terminal for only $325 and our CT-VM Monitor with matching cover for $175. Our MF-68 Dual Minifloppy costs just $995, complete with Disk BASIC and a disk operating system. For cassette storage our AC-30 Cassette Interface gives simple control for one or two cassette recorders.

You can get inexpensive hard copy with our PR-40 Alphanumeric Line Printer.

We back up the 6800 system with low-cost software, including 4K and 8K BASIC.

Compare the value you get with our computer and peripherals before you buy. We think you'll find that SWTPC gives you more for your money in every way.

Enclosed is:

_____ $995 for the Dual Minifloppy
_____ $325 for the CT-64 Terminal
_____ $175 for the CT-VM Monitor
_____ $395 for the 4K 6800 Computer
_____ $250 for the PR-40 Line Printer
_____ $79.50 for AC-30 Cassette Interface
_____ Or BAC # _____ Exp. Date_____
_____ Or MC # _____ Exp. Date_____
Name _____ Address _____
City _____ State _____ Zip_____

SWTPC Southwest Technical Products Corp.
219 W. Rhapsody, San Antonio, Texas 78216
London: Southwest Technical Products Co., Ltd.
Tokyo: Southwest Technical Products Corp./Japan

Our choice line of Personal Computing Books

NEW CHOICES

MICROPROCESSOR BASICS (Elphick) Selection and application info on 8 popular micro's, including the 8080, 6800, F8, IMP, and 6100. *#5763-6, paper, $10.95*

THE BASIC WORKBOOK: Creative Techniques for Beginning Programmers (Schoman) "Hands-on" learning of problem-solving using a computer. *#5104-2, paper, $5.50*

GAME PLAYING WITH BASIC (Spencer) Over 50 easy-to-learn and challenging games and puzzles for your personal computer. *#5109-3, paper, $6.95*

TELEPHONE ACCESSORIES YOU CAN BUILD (Gilder) Fully-illustrated, step-by-step instruction on building useful phone accessories at a fraction of the commercial cost. *#5748-2, paper, $4.95*

STANDARD DICTIONARY OF COMPUTERS AND INFORMATION PROCESSING, Revised Second Edition, *#5099-2, $16.95*

FUTURE CHOICES

APPLIED COMPUTING: Putting Your Computer to Work, *#5761-X, Available Jan. '78.*

PROGRAMMING THE PROGRAMMABLE CALCULATOR *#5105-0, Available Jan. '78*

CRITICS' CHOICES

MICROPROCESSORS: New Directions for Designers (Torrero) ". . . a useful book for the electronics design engineer." BYTE MAGAZINE. *#5777-6, paper, $10.95*

FUNDAMENTALS & APPLICATIONS OF DIGITAL LOGIC CIRCUITS (Libes) "A great book for use as a reference by people who are learning digital electronics." PEOPLE'S COMPUTER COMPANY. *#5505-6, paper, $7.95*

BASIC BASIC: An Introduction to Computer Programming in BASIC Language (Coan) ". . . an excellent introduction . . . clearly written and well-organized." COMPUTING REVIEWS. *#5872-1, paper, $8.95*

ADVANCED BASIC: Applications & Problems (Coan) "This one rates well above average." DATA PROCESSING DIGEST. *#5855-1, paper, $7.95*

#1

HAYDEN

in personal computing books!

AVAILABLE AT YOUR LOCAL COMPUTER STORE!

MORE CHOICES

GAME PLAYING WITH COMPUTERS, Revised Second Edition (Spencer) *#5103-4, cloth, $16.95*

COMPUTERS IN ACTION: How Computers Work (Spencer) *#5861-6, paper, $5.50*

COMPUTERS IN SOCIETY: The Wheres, Whys, & Hows of Computer Use (Spencer) *#5915-9, paper, $5.50*

PROGRAMMING PROVERBS (Ledgard) *#5522-6, paper, $6.95*

PROGRAMMING PROVERBS FOR FORTRAN PROGRAMMERS (Ledgard) *#5820-9, paper, $6.95*

COBOL WITH STYLE: Programming Proverbs (Chmura & Ledgard) *#5781-4, paper, $6.95*

MINICOMPUTERS: Structure & Programming (Lewis & Doerr) *#5642-7, cloth, $13.95*

DIGITAL SIGNAL ANALYSIS (Stearns) *#5828-4, cloth, $19.95*

FORTRAN FUNDAMENTALS: A Short Course (Steingraber) *#5860-8, paper, $4.95*

DIGITAL TROUBLESHOOTING: Practical Digital Theory and Troubleshooting Tips (Gasperini) *#5708-3, paper, $9.95*

DIGITAL EXPERIMENTS: Workbook of IC Experiments (Gasperini) *#5713-X, paper, $8.95*

COMPUTER MATHEMATICS (Conrad, Conrad, & Higley) *#5095-X, cloth, $13.95*

All prices subject
to change without notice.

HAYDEN BOOK COMPANY, INC., **50 Essex Street, Rochelle Park, New Jersey 07662**

SELECTRIC TERMINAL

- 14.8 cps print speed
- 13 inch line length
- 10 or 12 characters/inch
- 6 or 8 lines/inch
- IBM 2740-1, 2 line control
- IBM 2740 keyboard/printer

This deal can't be beat. Not only do you get a quality terminal for a fraction of the cost of anything else on the market, but you also get tyewriter quality. Can be interfaced to a mini- or microcomputer or used with IBM computers. These terminals are worth the price of a typewriter alone Our warranty is limited to replacing any defective parts for a period of 90 days following receipt of equipment. Interchangeable print element, of course. Includes attractive desk. Optional documentation package $25.00.

SUPER SALE

***JUST* $675**

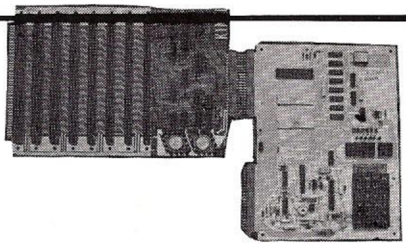
COMPUTER DEALERS ASSOCIATION

CALL [313] 994-3200 TO RESERVE

KIM MEETS S-100

Item	Kit Price	Assembled Price
Kimsi Backplane/S-100 Adapter 8 slots for S-100 available (requires +8V, +16V & -16V unregulated power supply)	$125 ☐	$165 ☐
KIM to Kimsi Connector Set Includes recorder cables	8 ☐	11 ☐
KIM-1 incl. documentation Send for more information	N/A ☐	245 ☐
S-100 Edge Connector Kimsi accepts 8; 1 is supplied	5 ☐	7 ☐
Cassette Recorder for KIM-1 Hundreds sold for KIM use	N/A ☐	55 ☐
4K Imsai RAM Board Bare minimum to run Tiny BASIC	139 ☐	239 ☐
8K Seals RAM Board Enough space for programs below	N/A ☐	269 ☐
Poly Video Terminal Interface 16 x 64 Connects to a Monitor & Keyboard	210 ☐	280 ☐
Matrox ALT-2562** Video Graphics Board displayed 256 x 256 array	N/A ☐	395 ☐
Itty Bitty Tiny BASIC Uses 2.5K; kit a paper tape, assembled a cassette	5 ☐	10 ☐
Focal [a DEC trademark] Includes floating point	N/A ☐	50 ☐
6502 Assembler & Editor This is NOT the MOS version	N/A ☐	40 ☐

SPECIAL! 10% off S-100 Boards when ordered with Kimsi.



Escape the single source blues. The Kimsi is the easy way to expand your KIM-1 system with readily available S-100 boards. Forethought Products has interfaced the 6502 to the S-100 bus by decoding the top 4K for IN and OUT instructions. Full capacity interfacing allows complete control of the 6502 and DMA from the S-100 bus.

Name ______________________
Address ______________________
City State Zip ______________________
Charge my BAC/VISA____ M/C____ Interbank # ________
Card # ______________________ Exp. Date ________
Signature ______________________
☐ Please send me info on the new Commodore PET 2001 computer

Total for goods checked above
+4% Michigan Residents
+4% Shipping and Handling
+$1 if order's under $20
Total Amount

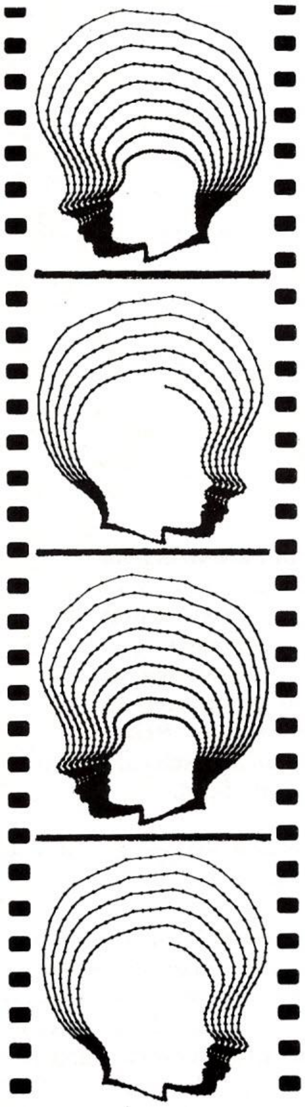
Limited 90-day Warranty
Return non-working merchandise within 90 days for replacement or refund. Kits are warranted to be complete with working components. All items subject to prior sale.

Newman Computer Exchange/CompuMart

1250 North Main Street Department **R1** P.O. Box 8610 Ann Arbor, Michigan 48107
(313)994-4445

Contents

Volume I, Number 6
December 1977



FEATURES

18 **Computers Challenge America's Cup** **by Eben Ostby**
Put a computer on board and what do you get? The racer's edge from DEC.

23 **A Beginner's Guide to Peripherals: Input/Output Devices Your Mother Never Told You About** **by Leslie Solomon and Stanley Veit**
Plus other goodies to ponder for your home computer.

30 **Computer Country: An Electronic Jungle Gym for Kids** **by Lee Felsenstein**
Running a computer center for fun and....

34 **The Best Slot Machine Game Ever** **by Tom Digate**
A game with a little life in it for your computer center—just in case you're thinking of opening in Atlantic City.

36 **The Micro Diet: Better Health through Electronics** **by Karen E. Brothers and Louise L. Silver**
Using your micro to keep track of the calories, vitamins, and minerals you cook up. Let the computer balance your diet as well as your budget.

43 **Come Closer and We Won't Even Have to Talk** **by Avery Johnson**
Communications has always been the tie that binds. Now dialogue between machines and people is becoming essential, and the crucial communications question may well be one of distance.

54 **The Kit and I, Part Four: Testing, Testing** **by Richard W. Langer**
The continuing saga of one man's struggle with soldering iron and components finally leads to a test pattern.

60 **Putting Two and Two Together: Part 0010** **by Tom Pittman**
Completing the beginner's guided tour through the world of binary arithmetic.

66 **Computer Models in Psychology** **by Joseph Weizenbaum**
Heuristics, information processing, physics, psychology—put it all together and what do you get? Artificial intelligence? Yes? No?

78 **Fair Play (A short story)** **by Steve McCue**
Computers come to baseball. But who calls the shots?

83 **Micro, Micro on the Wall, How Will I Look When I Am Tall** **by Stuart Dambrot**
Meet your future self in the CRT.

87 **Copycat Computer** **by Tom Digate**
A file-copy program for your personal program exchange.

90 **Talk Is Cheap** **by Hesh Wiener**
A quick rundown on talking to your computer and having it answer.

DEPARTMENTS

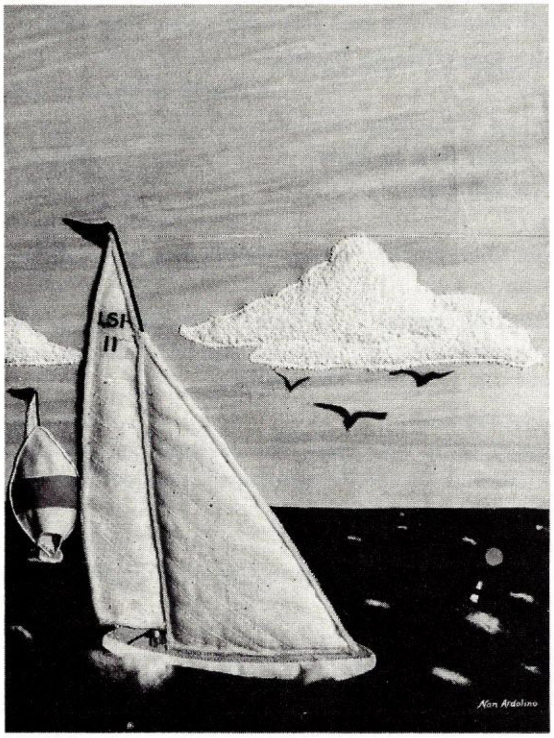
4 **On the Bus**
6 **Reader Interrupt**
42 **Eve 'n' Parity**
49 **Run On Micros**
50 **Centerfold**
96 **Babbage and Lovelace**
100 **PROMpuzzle**

COLUMNS

8 **Missionary Position by Theodor Nelson**
Personal Computing 1982
10 **The Human Factor by Dede Ely**
Who's at the Helm?
12 **AIQuotient by A. I. Karshmer**
Dr. Turing, I Presume?
96 **PROMqueries by Eben Ostby**
Have a Problem? Ask ROM
98 **Legal ROMifications by Peter Feilbogen**
Micro and Goliath
99 **FutuROMa by Bill Etra**
The Neutrino Connection

ROM is published monthly by ROM Publications Corporation, Route 97, Hampton, CT 06247 (Tel. 203-455-9591). Domestic subscriptions are $15 for one year, $28 for two years, and $39 for three years. Canada and Mexico $17 for one year, $30 for two years, and $41 for three years. For European and South American subscriptions, please add $12 per year additional postage. For all other continents, please add $24 per year additional postage. *Copyright © 1977 by ROM Publications Corporation. All rights reserved.* Reproduction in any form or by any means of any portion of this periodical without the written consent of the publisher is strictly prohibited. The following trademarks are pending: AIQuotient, Babbage and Lovelace, Eve 'n' Parity, floppyROM, futuROMa, The Human Factor, Legal ROMifications, Missionary Position, The Noisy Channel, On the Bus, PROMpuzzle, PROMqueries, Reader Interrupt, ROMdisk, ROMshelf, ROMtutorial, and Run On Micros. Opinions expressed by authors are not necessarily those of *ROM* magazine, its editors, staff, or employees. No warranties or guarantees explicit or implied are intended by publication. Application to mail at second-class postage rate pending at Hampton, CT 06247. Membership in Audit Bureau of Circulation pending.

On the Bus



Nan Ardolino, a recent graduate of Parsons School of Design, is a free-lance illustrator. Ardolino's unique use of material has created a new textural dimension. The fabric construction is appealing to the sense of touch as well as vision, simulating an indefinable third dimension when reproduced in printed form as on this month's cover of *ROM.* Black and white work, watercolors, and pastels, are also media of which she is particularly fond.

Hooked on crossword puzzles at an early age, **Daniel Alber** now constructs as well as solves them. Part of the Brownstone Renovation generation in New York, when he's not constructing puzzles for the likes of *Field and Stream, The New York Times,* and *ROM,* he's reconstructing olden golden rooms in his Brooklyn based house.

Karen E. Brothers and **Louise L. Silver** were college roommates at M.I.T. Karen has an S.B. in mathematics and two children, and lives in Wayland, Mass. Louise has an S.B. in political science and two children also, and lives in Arlington, Mass. Their computer consulting company is called Consultus.

Stuart Dambrot is currently a student at the University of Connecticut. Disillusioned with traditional psychological theories, he has become interested in computer science and artificial intelligence. Stuart is now trying to develop alternative hypotheses about the nature of human cognition.

Tom Digate is an IEEE member who enjoys playing chess—with humans or computers. He decided to get his own computer after reading the now historic *Popular Electronics* article on the first Altair. Currently, he is building an interface for wedding his Selectric and Sol into a happy hard-copy system.

Dede Ely is a free-lance writer and critic who lives on Cape Cod. Basically, she prefers sailboats to computers.

Bill Etra is a West Coast based computer design consultant. He is co-inventor of the Rutt/Etra Video Synthesizer—the first portable voltage control analog video synthesizer, as well as the Videolab. His main interest is videographics, and many of his works have appeared as cover illustrations on various periodicals and books including *Computers in Society* and *Broadcast Management and Engineering*. His current research centers on "The Computer as a Compositional Tool for Video."

Peter Feilbogen attended the Rutgers School of Business Administration and Brooklyn Law School. In addition to being an attorney, he is also a Certified Public Accountant. He has been a sole practitioner on Long Island for approximately ten years, and treasurer of Data Information Services, Inc. An avid tennis player, he is currently trying to improve his game with the aid of a computer.

Lee Felsenstein was born in Philadelphia and grew up wanting to be an inventor. Outside of that, he bears no resemblance to W. C. Fields whatsoever. Instrumental in establishing the first experimental public-access information-exchange system in 1972, he is presently engaged in further development in that area of communications. In his spare time he has designed the Pennywhistle 103 modem, the VDM-1 video display module, the Sol terminal/computer, and the VID-80 video display card. Lee was also instrumental in forming the original Homebrew Computer Club and currently serves as its "toastmaster."

Avery Johnson, with a doctorate in electrical engineering from M.I.T. and five years of post-graduate study in neurophysiology, is uniquely qualified to aid in the goal of making "man-relevant engineering a viable way of living and working." His work is with mobility and sensory aids for the blind, communications systems for crippled children, and physiology of communication. He is currently pursuing research in "soft control material" to provide new interfacing technology between man and machine.

A. I. Karshmer is currently completing his Ph.D. in computer science at the University of Massachusetts. His main interest is the use of artificial intelligence concepts in solving problems involved in the transmission of computer graphics. Currently, he is developing a method for sending high-density information, such as animated graphics, over existing low-bandwidth telecommunication networks.

Richard W. Langer is a free-lance writer whose articles have appeared in such diverse magazines as *New York, Family Circle, House and Garden,* and *Esquire.* Currently he is a columnist with *The New York Times* and at work on his fifth book.

Steve McCue works in the data transmission department of the Bell System. He has written for a number of periodicals, including the *Bell System Review.* His main writing thrust is fiction. Having written numerous short stories, he is now halfway through a novel based on computers and corporate life.

Theodor Nelson is the author of the classic *Computer Lib/Dream Machines,* a Whole Earth style catalogue of computer machinations. Presently at the department of mathematics of Swarthmore College, where he is working on the Hypertext Project, Ted specializes in highly interactive systems for graphics and text. His past experience includes a stint at Dr. Lilly's Dolphin Laboratory and work as a consultant for Bell Lab's ABM system.

Eben Ostby first began experimenting with computers while at the Pomfret School, which had a PDP-8. He went on to become the first computer science major at Vassar College, from which he recently graduated with honors. Ostby has done extensive work with graphics in APL, and recently programmed what he thinks may be the first cartographic project in APL. Currently, he is a graduate student at Brown University.

Tom Pittman's dream, ever since high school, was to own a computer. Well, now he does—in fact he owns several. Starting with an Intel prototype 4004 system in 1972, he has branched out, turning his hobby into a business, albeit a tiny one, as the main implementor of Tiny BASIC.

Leslie Solomon is Technical Editor of *Popular Electronics,* where he devotes half his time to computer developments. He introduced the first personal computer, the Altair, and later many others including, most recently, Speechlab, the first vocal interface for a personal computer.

Stanley Veit has been in the computer field for about twenty years, originally as a technical writer and instructor of computer maintenance. He is co-author (with Leslie Solomon, Technical Editor of *Popular Electronics*) of *Getting Involved With Your Own Computer.* Veit founded the first computer store on the East Coast.

Joseph Weizenbaum is Professor of Computer Science at the Massachusetts Institute of Technology. He is best known to his colleagues as the composer of SLIP, a list-processing computer language, and for ELIZA, a natural-language processing system. More recently, he has directed his attention to the impact of science and technology—and of the computer in particular—on society.

Hesh Wiener is the editor of *Computer Decisions,* the most widely read computer trade publication in the United States. Previously he was on the academic staff of the University of California at Berkeley, where he headed the Computer Education Project at the Lawrence Hall of Science. During that time he spent a year teaching blind children to use computers. He designed and built some of the equipment used by the blind students.

A Note to Authors:

ROM is always looking for good computer applications articles from people with up-and-running systems. We also will be glad to consider for possible publication manuscripts, drawings, and photographs on other computer-related subjects. Manuscripts should be typewritten double spaced, and a stamped self-addressed envelope of the appropriate size should accompany each unsolicited submission. Although we cannot assume responsibility for loss or damage, all material will be treated with care while in our hands. Manuscripts should be sent to ROM Publications Corporation, Route 97, Hampton, CT 06247.

RAINBOW COMPUTING, INC.

Supplier of

WAVE MATE
THE DIGITAL GROUP
DEC PDP
Computer products

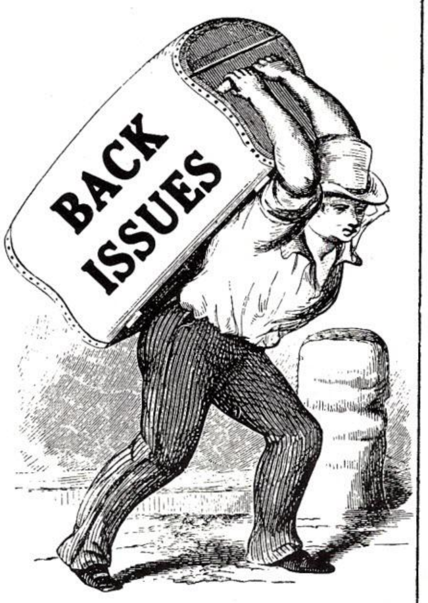
Peripherals and Supplies from

PERSCI
CENTRONIX
DIABLO
MAXELL
COMPUTER DEVICES
LEAR-SIEGLER
MULTI-TECH
TEXAS INSTRUMENTS

Specialists in Design, Implementation
and Support of
Custom Hardware/Software Systems for
Business, Educational, and Personal Use

Experts in most major computer
software including
CDC, IBM, PDP
BASIC, COBOL, FORTRAN, PL1
LISP, SIMULA, SNOBOL, SPSS, BMD's
COMPASS, MACRO,
6800, & Z80 assembly languages

10723 White Oak Ave., Granada Hills,
Ca. 91344
(213) 360-2171



Complete your collection
NOW!
Back issues $3.00 postpaid from
ROM Publications Corp.
Route 97
Hampton, CT 06247

ROM

Reader Interrupt

Dear ROM,

Please note two errors on page 62 of my article "Scott Joplin on your Sci-Fi Hi-Fi" (October issue of *ROM*):

1. For "C = 220Hz" I should have had "A = 220Hz".
2. "n = # of semitones above 'middle C' = 262Hz" should be "n = # of semitones above A = 220Hz".

Thank you.

Dorothy Siegel
Brooklyn, New York

ROM regrets letting the error slip by. Then again the editor is rumored to be tone deaf.

Dear ROM,

I have just read through "Tools" by Joseph Weizenbaum (in *ROM* Volume I, Number 1) in the first issue ever of your marvelous publication. This is a stunning and exciting essay, and it has put me onto a very serious question that I hope *ROM,* or Weizenbaum, or someone with more understanding of the computer than I have will pursue.

The question has to do with the presumed calculations, most of them computer-based, for the safety of nuclear power plants, as presented especially in the famous (or notorious) Rasmussen report. In the line of journalistic duty last year, I interviewed Richard Webb, author of *The Accident Hazards of Nuclear Power Plants* (Univ. of Mass. Press, 1976). Webb claims, and his book supports the claim in detail, that the Rasmussen calculations are based on at least seven questionable assumptions or "working guesses" open to criticism, and implies that these assumptions have become part of the functioning computer lore on which nuclear power decisions are based. Webb says his own calculations, inhibited as they are by his lack of access to a computer—an interesting kind of political and scientific dilemma in itself—suggest far different results than those of Rasmussen's team, and far scarier results at that. The computations necessary to assess nuclear power hazards scientifically would not only tax the capabilities of the nation's computers, but are discouraged by vested interests, according to Webb. In addition, he claims that nuclear power promotion by the government is unconstitutional, but that argument is basically an extension of his more important one that nuclear power development is the result of a cultish, computer-based mythology that does not put its beliefs to the test of real experimentation.

Is there any way of assessing the use of the computer as a "tool" in the deliberations over the safety and, on the other hand, purported benefits of nuclear power? Has computer potential been abused to satisfy a myth of nuclear power progress or have the safety analyses been in fact eminently sane and objective? And is Webb correct in his estimation of the sheer quantity of the computer work that would be required to take this argument once and for all out of the realm of second-guessing and all that horrible bribery that goes on in the name of research grants?

I sincerely hope that ROM will pursue these questions—for me they are among the ultimate ones regarding computer applications to living.

Robert Abel
Lake Pleasant, Massachusetts

Dear ROM,

I should like to take this opportunity to express my appreciation for the series by Joseph Weizenbaum. I noted in the letters column of October an objection to articles of the nature "The general theory of..." and decided to cast a vote for Professor Weizenbaum lest someone on your staff believe there is no reader interested.

The only other comment I have is on the article YOU, INC.? by Peter Feilbogen. "Subchapter S" corporations are not all "garage-type" manufacturing operations. The Anderson Corporation in which I hold stock is a "Subchapter S" corporation with sales in excess of $10,000,000 annually. The size plant involved in such a corporation is less relevant than the number of shareholders and the class of stock.

Robert D. Deane
Santa Monica, California

ROM
COMPUTER APPLICATIONS FOR LIVING

Editor and Publisher
Erik Sandberg-Diment

West Coast Editor
Lee Felsenstein

Associate Editors
Sue Neillson
Janet C. Robertson

Assistant Editor
Karen K. Jambeck

Contributing Editors
Sandra Faye
Bill Etra
Louise Etra
Ed Hershberger
Avery Johnson
Arthur Karshmer
Richard W. Langer
Theodor Nelson
Robert Osband
Eben Ostby
Frederik Pohl
Andrew Singer
Alvin Toffler

Crossword Puzzle Editor
Daniel Alber

Editorial Assistant
Donna Parson

Art Director
Susan Reid

Contributing Artists
Steve Gerling
Robert Grossman
Cindy Hain
Luis Jimenez
Korkie
Rex Ruden
Linda Smythe

Staff Photographer
Thomas Hall

Composition
Lynn A. Archer

Special Assistant
Jennifer L. Burr

Counsel
Peter Feilbogen

IT TOOK THE MEDIA TO KEEP THE GOVERNMENT HONEST. BUT IT TAKES "MORE" TO KEEP THE MEDIA HONEST.

MORE watches the media while the media watches you

What if the news reporters, TV commentators, gossip columnists and media gurus who helped write the Watergate story did as thorough a job investigating their own business?

What if Woodward and Bernstein found a Deep Throat somewhere in the bowels of the Times?

Or if Barbara Walters put Barbara Walters under the microscope?

That's the kind of thing that happens each month in the pages of MORE, the Media Magazine.

MORE covers the media like the media itself covers a big story. By looking for sources and listening and digging and watching every word and reading between the lines. By getting behind the scenes and into the back rooms and conference rooms, providing the stories behind the stories you get and the stories behind the stories you never get.

At MORE, we get media people to tell us things they'd never tell anyone else. And we get the very people who report, comment and advertise to write things about reporting, commentating and advertising they could never write anywhere else.

For example we've explored a family feud at the *Times* that may have been responsible for Daniel P. Moynihan becoming a U.S. Senator. We examined the power of a handful of editors at *Time* and *Newsweek* to create and destroy rock stars overnight. We interviewed the controversial *Los Angeles Times* reporter who said that journalists should "lie, cheat, steal or bribe to get their story." And MORE ran the Nora Ephron media column that *Esquire* killed and the profile of Rupert Murdoch that *New York* wouldn't run.

We've given the business to the news business, advertising, movies, publishing and the entire communications industry. And don't think they haven't started watching their words a little more closely now that they know someone else is.

So if you subscribe to the idea that someone should be watching the media like the media watches everyone else, subscribe to MORE.

I WANT TO KEEP THE MEDIA HONEST. SEND ME MorE.

Please enter my subscription immediately.

☐ $12 for 1 year (12 issues)
☐ $21 for 2 years (24 issues)
☐ $30 for 3 years (36 issues)

☐ Payment Enclosed ☐ Bill me

Name ______________________________

Address ______________________________

City ______________ State ________ Zip ________

Signature ______________________________

MorE THE MEDIA MAGAZINE P.O. Box 955 Farmingdale, New York 11735

Add $2.00 per year for outside U.S. and Canada. Please allow up to 4 to 6 weeks for delivery of first issue.

MR57

Missionary Position

PERSONAL COMPUTING 1982

by Theodor Nelson

Well, 1982 certainly is an exciting year for personal computing. And Atlantic City in August, with gambling, is the same as it was but more so, if you can stand it.

America is computer crazy. Women of fashion wear silver pants suits full of twinkling lights and cunningly placed switches. Tully Peschke has become America's top fashion model, edging out Margaux and Farrah to model the new Hot Solder line of cosmetics, perfumes, and "component" jewelry.

There has been some social upheaval, but thanks to several new social-service agencies—especially, those offering rehabilitation for COBOL programmers—things have settled down.

Well, there sure have been a lot of changes in the industry.

The new ten-megabyte memory board from MITS has struck many as a disappointment. Considering the cost—a hundred dollars—they could have offered a lot more. However, a spokesman for their subsidiary, PERTEC, says, "This is just the beginning."

Processor Technocracy—a firm merging the old Processor Tech with its recent acquisition, Univac—has conservative plans. Their new desktop computer, SOLOMEN, only offers two megabytes of sixty-four-bit memory. "But we believe we stand for quality," says corporate vice-chancellor Steve Dompier.

Crothers Memorial Corporation (formerly Cromemco) has brought out its HyperDazzler: one million points square, 128 colors, with as many local CPUs controlling subsegments of the picture. They admit that widespread use in schools will have to wait a year or two.

Ohio Scientific has announced a new computer containing one of every processor chip, together with a compiler optimized to use them all in parallel, even when not needed.

IMSAI has just announced a radical new computer design called the Hypercube. How to program it is a mystery, but it promises many operations per second. Observers are puzzled by the press releases, which seem to have earlier dates crossed out.

The increasing number of IBM 370s coming onto the hobby market has raised the question, "What is the best way to hook up a 370 for hobby use?" Ignoring the obvious unprintable answers, several interfaces are available. Intel, despite its add-on product line for the 370, has staunchly refused to interface the 370 to the S-100 bus, which they call illegitimate (not the precise term), despite the three thousand interesting peripherals now available. The interface has, however, been supplied by Itel, not to be confused with Intel.

Another S-100 adapter for the 370, however, simply uses the 370 as a peripheral, and many users say this works best.

People's Computer Company has changed its name to Transamerica Peoplecomp, partly because of it being recently acquired, and partly to avoid its being confused with People's Computer *Commune* (formerly IBM Federal Systems Division). Transpeep, as they like to be called, has just introduced a new magazine: *Mrs. Tillie Mapes' Newsletter of Flying Elephants, Funny Green Snakes and Muffins,* which, of course, deals with hardware maintenance of S-100 products.

A second edition of *Computer Lib* is rumored to be in the works.

International Honey Control (a firm merged from IBM's Advanced Systems Development Division, Honeywell, and Control Data) has announced that it will go after the business market for large systems, whatever that may be. Meanwhile, Interplanetary Godco—a consortium effort of Intel, IBM Components Division, and Godbout Electronics—is tooling up for the production of HLIC (Humungously Large Integrated Circuits) in satellite orbit. (Speaking of mergers and acquisitions, DEC rejected a tender offer from Technical Design Labs. They rejected it even when the offer was made less tenderly. "We stand by our growth rate," says doughty Ken Olsen, who was actually standing by a stone wall as he made the remark.)

The scene in software remains in flux. Languages once thought brilliant, like SMALLTALK, FORTH, and PASCAL, have come and gone. But they have left their mark. Most versions of BASIC now have procedures, if-then-else, user-definable data types, and dynamically modifiable arrays of arrays of arrays of variable length. But people are increasingly attracted to the new languages, like R2D2, HYPERWUMPUS, GLITZ, PEEKABOO, and SNERD.

The autodialling accessories have created a new scourge in the society, nuisance telephone calls made by computers. These are used alike by politicians, advertisers, and callers who play tapes of heavy breathing. (In one strange case, the answering party hears *really* heavy breathing—followed by a strange voice saying, "This is a dragon!")

A happy event for the nation has been the surreptitious cancellation of the National Debt by government programmers. When Donn B. Parker vowed he would find the culprits, he found himself appointed Ambassador to Liechtenstein.

Well, the conference has been a smashing success. Though competing with Wayne Green's "Computer Machismo" conference in Acapulco, the Convention has still managed to draw a hundred thousand people. Incidentally, Green's new magazine, *Wire Rap,* has had some success in interesting everyday computer owners—housewives, secretaries, schoolchildren—in learning about hardware assembly and principles.

Yes, computer hobbyism has come a long way. Unfortunately, this Convention was, as usual, closed to those having prebuilt computers—approximately forty million by this time. And in the next few months that number is expected to double, with Kellogg's inclusion of a computer in every cereal box. (They expect to get you on the coupons for peripherals.) ▼

You're curious enough to read ROM now, you'll be curious enough to read ROM every month. So subscribe!

SAVE $ 7.00 over the single-copy price
with a one-year subscription

SAVE $20.00 over the single-copy price
with a two-year subscription

SAVE $33.00 over the single-copy price
with a three-year subscription

ROM
COMPUTER APPLICATIONS FOR LIVING

ROM Publications Corp.
Route 97
Hampton, CT 06247

Name ____________________

Address ____________________

City __________ **State** __________ **Zip** __________

United States: ☐One year $15 ☐Two years $28 ☐Three years $39
Canada and Mexico: Please add $2 per year additional postage
Europe and South America: Please add $12 per year additional postage
All other continents: Please add $24 per year additional postage
☐**Check or money order enclosed** ☐**Master Charge** ☐**BankAmericard**

GUARANTEE:
If not satisfied with **ROM** at any time, let us know. We'll cancel your subscription and mail you a full refund on all copies still due you.

Please allow 4-6 weeks for delivery. **Exp. date** __________ **Card#** __________

If you wish to be billed, please use either Master Charge or BankAmericard. Direct billing by the publisher is the single largest cause of subscription service problems. With today's postage rates, it's also very expensive for you as well as for ROM.

The Human Factor

WHO'S AT THE HELM?

by Dede Ely

Several months ago, Andrew and I began our day at Newport with a debate over which excursion boat to pick for watching the initial race of the America's Cup series. *Courageous* was the odds-on favorite against *Australia.* Newport was a chaotic and crowded scene, and we were two among many trying to pick a winner of our own that morning. Our choice was between a small ferry which looked agile and maneuverable or another larger ferry which looked dry, comfortable, and seaworthy. When faced with a long day at sea, we opted for the larger, more comfortable vessel and boarded it.

As we set off from the dock and steamed full speed ahead out of the harbor, I paced the observation decks with my ears like pitchers. Since this was our first Cup adventure and since Howard Cosell was nowhere to be either seen or heard, my strategy was to eavesdrop on the prerace grapevine until I located a knowledgeable person. I thought we could then position ourselves near him for the best possible tack-by-tack analysis of the race. As I wandered here and there, the bits and pieces began to add up to zero. Clearly no one knew much more than I did, and we were on our own.

We settled for borrowing a program and jockeying for the best viewing position at the bow. With luck and persistence we ended up very near the bow where we could see action to both starboard and port. Ironically, just as we had given up looking for an expert, a nautical-looking young man, who, I inferred, worked for a yachting magazine, squeezed in behind us. Furthermore, he had as his guest an Australian, a non-sailor at that. We were set.

After crashing into four-foot waves for more than an hour, passing and being passed by small sailboats and Navy cruisers in a steady stream of vessels leaving Newport, we arrived at the starting line. There we saw an incredible flotilla of every imaginable vessel, from opulent yachts with Italian names out of Chicago to small, salty sailboats from snug New England harbors, from Dyer Dinks to the Coast Guard cutters marking the race course. Private airplanes, helicopters, and even a blimp drifted overhead.

Our captain immediately revealed his cautious nature. He was maddening or, as we heard one man mutter, "a total timid turkey." Afraid of risking a collision, he refused to bring us up close to the starting line. He consistently kept back from the action and, even when our speed allowed us to arrive first at a critical buoy, he held us back in a poor viewing position and allowed other boats to get in our way. Eventually the passengers started booing and hissing and were soon yelling instructions up to the wheel room.

At the sound of the starting gun, the "twelve-meters" shot over the line and *Courageous* took the lead by two seconds. They were off on a tacking duel into the wind and were soon out of sight of the spectator boats. The flotilla headed off on a dogleg course to rendezvous at the first marker. It was time for a bite of lunch and, for what I cynically surmised was the point of the trip for many on board, the first drink of the day.

Soothed by the sea, I settled back into a deck chair and tried to catch an idea that was floating around the perimeter of my mind. What was this experience reminiscent of? Hmm. Yes. The Personal Computing Conference in Atlantic City we had been snookered into several weeks earlier.

Not that Atlantic City is like Newport. Any resemblance between the two stops with their proximity to the Atlantic Ocean. Newport is a charming, old nautical town which is currently undergoing a renaissance. The kindest, and the most accurate, epithet I heard hurled at Atlantic City all weekend was "rat hole." The conference planners should have their fingers pricked to bleeding with the pin they used to mark the map for this one. Tsk, tsk.

At any rate, at the conference I had had the same feeling as when I walked around the decks of the ferry, looking for an expert. It was a feeling that no one around me quite knew what was happening. I think it true of the personal computing field in general. No one knows quite where it is going or why.

Take as an example the question of applications, a question that no one in personal computing likes to confront. What will people, hobbyists, do with their minis and micros? No one seems to have progressed much beyond game playing. We saw razzle-dazzle *Solitaire, Monopoly,* and *Blackjack* on color video screens at the conference, but meaningful applications were rare. In our house we have discussed checkbook balancing, personal finance accounting, recipe files, and even an automated greenhouse, but all of these uses would, at present, be more troublesome than the problems they would supposedly solve.

And who will solve these problems? Who, or what, are

DON'T FORGET THE DIGITAL FOAM CONTEST

Beside our regular payment to contributors, we will be giving away two prizes for the best applications article on digital foam we receive before January 1, 1978.

First prize is your choice of $500 or its equivalent in Dynacon. Second prize is $250 or its equivalent in Dynacon.

The articles should be 1,500-3,000 words long. They should include diagrams (roughs are fine) and photographs of your system in operation. You must be able to demonstrate that you have it up and running.

For further details see *ROM,* Vol.I, No.1.

Dynacon material kits for experimentation and implementation are available from:

Dynacon Industries Inc.
14 Bisset Drive
West Milford, N.J. 07480

for $10 if you send a check with your order, or $12 if you wish to be billed.

the hobbyists? At the conference, the diverse nature of this group was clear. A random sampling of conferees produced a stereotypical eighteen-year-old, acne-ridden technical wizard who had assembled an entire minicomputer system in his garage with "el cheapo" discount components, used equipment, and rock-bottom kits, which he loved because they were "so hard to use." There were also a pair of businessmen, who were interested in industrial applications for micros but bewildered about where or how to start; ham and CB radioers galore; programmers on bus-persons' holidays; and the inevitable little old lady in tennis shoes who "just wanted to know what all the brouhaha was about." At least in Newport we could tell the racers from the spectators.

Next there was the question of how to pick a winner. The personal computing field is fragmented. Small companies come and go, usually at night, offering a myriad of plug-in compatibles, bells and whistles, "parasitic" add-on improvements, and what have you. Authors write hair-raising articles about misassembling computer kits and about the pros and cons of the various tiny languages.

At the PCC, as on our ferry boat which listed to whichever side the action was on, curious onlookers consistently crowded around the booths of Heathkit, Radio Shack, and Commodore. At Newport it seemed clear after the first two seconds of the race who would be the victor, but after the PCC, the big question still remained. Who will be the IBM of personal computing, or as they say in Newport, "Will the real Captain Courageous please stand up?"

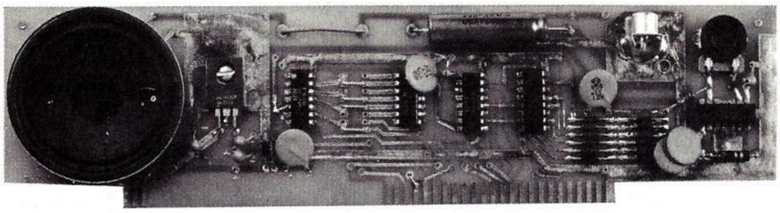
Now that the Big Guys have entered the scene, everyone is looking to them to find a winner. Will their size alone enable them to bring computers closer to the people? Possibly. But I think it more likely that, like the maddening captain of our large ferry boat, they will be afraid to take the risks that will bring them close to the action.

Playing Coast Guard to the flotilla of exhibitors and conferees at the PCC were the magazine publishers. *ROM, BYTE, dr. dobb's,* and *Interface Age* are all attempting, with varying degrees of success, to be part guru, part technology tutor, part prophet, and part sheik in the bazaar of technology. Unlike the Coast Guard, however, the publishers aren't quite sure who is in the race or, sometimes, where the course is. And while they may know what the bottom line is, the whereabouts of the finish line remains elusive to all of them.

Twelve-meter yacht racing, like computer hobbying, is a high-technology sport. After the race when *Courageous* had defeated *Australia* by more than a minute, we stopped at the pier to see her hauled from the water. Her hull looked more like a supersonic jet wing than the bottom of a sailboat, and I realized just how much the twelve-meter racing game is a product of the application of twentieth-century technology to an ancient art.

Computing, too, represents an ancient art that is being advanced by modern technology. And with computers, as with yachts, it is the human that makes the difference. My prediction is that the computer that reaches out and is accessible to people will find and cross the finish line well ahead of the competition. Both America's Cup syndicates spent millions on their contenders, but in the final analysis the difference was in the captain at the helm, the human factor. ▼

NEWTECH
Model 6
MUSIC BOARD



PRODUCES MELODIES, RHYTHMS,
SOUND EFFECTS, MORSE CODE,
TOUCH-TONE SYNTHESIS, AND MORE!

- S-100 bus compatible
- Jumper selectable address decoding
- 6-bit latching digital-to-analog converter
- Glass epoxy printed circuit board with plated-through holes and gold-plated fingers
- Audio amplifier
- Speaker

$59.95 ASSEMBLED AND TESTED
AVAILABLE THROUGH YOUR
LOCAL COMPUTER STORE

- Volume control
- RCA phono jack for connection to external audio system
- Complete users manual with BASIC program for writing musical scores and 8080 Assembly Language routine to play them
- 60-day parts and labor warranty

NEWTECH COMPUTER SYSTEMS, INC.
131 JORALEMON STREET * BROOKLYN, NEW YORK 11201 * (212) 625-6220

AIQuotient

DR. TURING, I PRESUME?

by
A. I.
Karshmer

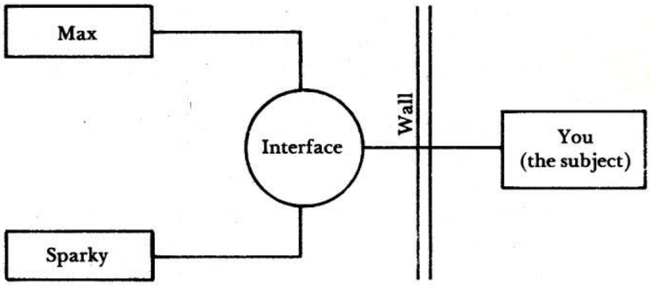
You are sitting in a small quiet room—the only sound you hear is the hum of the fan in your computer terminal. The terminal is connected to a wire which leads out of the room into some sort of interface which will act as a message switcher between you and some unknown source of data. Suddenly your terminal comes to life and the following message appears before you:

> *HELLO, MY NAME IS MAX. I AM A HUMAN BEING COMMUNICATING WITH YOU VIA THIS TERMINAL FROM ANOTHER ROOM. ALL OTHER MESSAGES YOU RECEIVE FROM OTHER SOURCES ON THIS TERMINAL ARE GENERATED BY A REMOTE COMPUTER AND NOT BY A HUMAN BEING. PLEASE IDENTIFY YOURSELF AS I LIKE TO KNOW WITH WHOM I AM COMMUNICATING.*

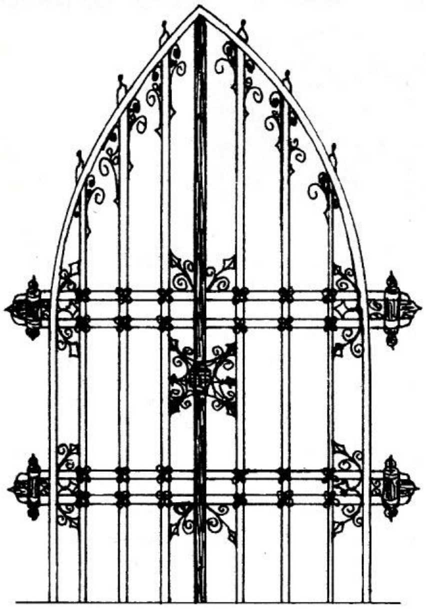
No sooner do you type in your name and press the carriage return key than the following message appears on your terminal:

> *HELLO, MY NAME IS SPARKY. I AM A HUMAN BEING COMMUNICATING WITH YOU VIA THIS TERMINAL FROM ANOTHER ROOM. ALL OTHER MESSAGES YOU RECEIVE FROM OTHER SOURCES ON THIS TERMINAL ARE GENERATED BY A REMOTE COMPUTER AND NOT BY A HUMAN BEING. PLEASE IDENTIFY YOURSELF AS I LIKE TO KNOW WITH WHOM I AM COMMUNICATING.*

Once you identify yourself to Sparky, you are then encouraged by both sources to carry on any sort of dialogue, in English, with either or both of the two other participants. The ultimate goal of the communication process is for you to determine which of the two sources, Max or Sparky, is indeed the human being and which is the computer.



The situation described above is, in fact, the Turing Test of artificial intelligence. The test was formulated by A. M. Turing, a famous mathematician who carried out


OPEN A GATE FOR SOME FRIENDS

Send a gift subscription to ROM today

ROM
COMPUTER APPLICATIONS FOR LIVING

Please send a gift subscription to:
Name
Address
City State Zip
United States: ☐One year $15 ☐Two years $28 ☐Three years $39
Canada and Mexico: Please add $2 per year additional postage.
Europe and South America: Please add $12 per year additional postage.
All other continents: Please add $24 per year additional postage.
☐Check or money order enclosed ☐Master Charge ☐BankAmericard
Exp. date Card#
Name of sender
Address
City State Zip
☐I would like a gift card enclosed with the following message:

ROM Publications Corporation, Route 97, Hampton, CT 06247

pioneering research in such areas as computer logic, undecidability theory, and artificial intelligence. Turing is probably best known for his Turing machine (not a small Italian sports car) which in many ways was the theoretical foundation for all modern computer science. Turing's test of artificial intelligence basically states:

> If the subject (in this case, you) cannot determine which of the two unknown sources is a computer and which is human with significantly better than fifty percent accuracy, and if this result continues to hold regardless of which people are involved as subjects in the experiment, the machine is said to simulate human intelligence.

While Turing's test has never been seriously attempted, and for that matter may never be, it does create a simple intuitive measure of what it means to build a machine with a high degree of artificial intelligence (AI). The following notes on the test will help us further understand some of the key issues in the field of AI:

1. The subject is not allowed to observe the physical characteristics of the two other test participants (the human and the computer). He is only permitted to observe their intellectual behavior—their ability to communicate and to think in an abstract fashion. This of course means that the test subject is not able to observe the ability of the other two participants to function in real world, nonabstract situations. Thus, the ability to carry out such tasks as driving a car or changing a baby are not measures of artificial intelligence according to Turing. (It is important to note here that research into creating machines to carry out such real world tasks is indeed a central focus of much AI research currently being carried out.)
2. A possible theoretical outgrowth of the Turing Test would be to create a complete mathematical description of a machine that could pass such a test of intelligence, or, conversely, to prove that no such machine could be built. (Clearly, a complete proof of the existence of such a machine is as valid as actually building the machine.)
3. As an extension of note 2, the question must be asked, "Is there more than one type of machine that could pass the test?" And, if there is indeed more than one type of machine, is it possible to describe a class of machines which is capable of the task?
4. Finally, if it is possible to create a machine that can simulate human intelligence to the extent that it is capable of learning and problem solving, is it then possible to create a machine which possesses greater than human intelligence?

While the artificial intelligence research community has not seriously attempted to solve the whole problem of simulating human intelligence, it has attempted to come to grips with some of the smaller subproblems which are in some senses more interesting than the overall solution. In

FOR THE NUTRITION-CONSCIOUS COMPUTERIST

from

consultus

Now you can let your home computer check your daily diet—mother's favorite recipe never had it so good.

NUTRIVALUE I

Nutrivalue I supplies the conversion factors and nutritional data on over fifty basic ingredients. The data has been built into the program in the form of DATA statements. It can be ordered as a complete listing, with or without 8 level ASCII paper tape. Installation instructions and ingredient code chart are included. The additional DATA statements require about 3K memory, bringing the program size to about 5K. Users lacking adequate space can selectively remove items from the data set (instructions are given).

NUTRIVALUE II

Consultus also offers a more powerful analysis program with the following features:

* Ingredients can be entered by name
* Measurements (cups, etc.) can be entered by name
* Quantities accepted as mixed fractions (1 and 1/3)
* Expanded food item data set, outside of program
* Seventeen nutrients analyzed, including cholesterol and fiber
* Improved output format

Mail to: Consultus, PO Box 86, Arlington, MA 02174

Note: NUTRIVALUE I and NUTRIVALUE II are registered trademarks of Consultus.

☐ Please send 50-ingredient data set for the NUTRIVALUE I program.
Enclosed is ☐$10.00 for listing. ☐$13.00 for listing and ASCII paper tape.

☐ Please send information about the more powerful NUTRIVALUE II program and larger data sets.

Name____________________

Address________________ City__________ State______ Zip______

Computer configuration____________________

following issues of *ROM,* I will attempt to present and explain some of the research being carried out in a wide range of artificial intelligence research areas. Some of the areas I hope to cover are:

Game playing by computer

- Checkers and in particular Samuel's Checker Player, which is capable of learning to play championship level games
- Chess playing programs and the International Computer Chess Tournament
- Bridge playing programs
- *GO* playing programs
- General game playing programs

Robotics—how they work and what they can do

- SHAKEY and other high level robots which are capable of interacting with their environment
- Industrial robots such as those under development at General Motors and several industrial companies in Japan
- Home robots which are becoming available in the United States

Pattern recognition systems

- Handwriting analysis systems like those being developed by the United States Postal Service to aid in mail sorting by zip code
- Scene analysis systems such as those being developed for the Department of Defense (One of the more advanced systems is now under development at the University of Massachusetts.)
- Picture enhancement systems like those used by the National Aeronautics and Space Administration to enhance pictures coming back to Earth from space probes
- Speech understanding systems currently under development, and, in particular, the research being carried out at Carnegie-Mellon University.

Theorem proving

- General techniques
- Heuristic search strategies
- Reasoning by analogy
- Applications of theorem proving in automatic programming

Semantic information processing

- Understanding natural languages by computer—one of the early hoped-for uses of computers which never materialized
- Question answering systems
- Goal directed knowledge systems such as those used in conjunction with scene analysis systems

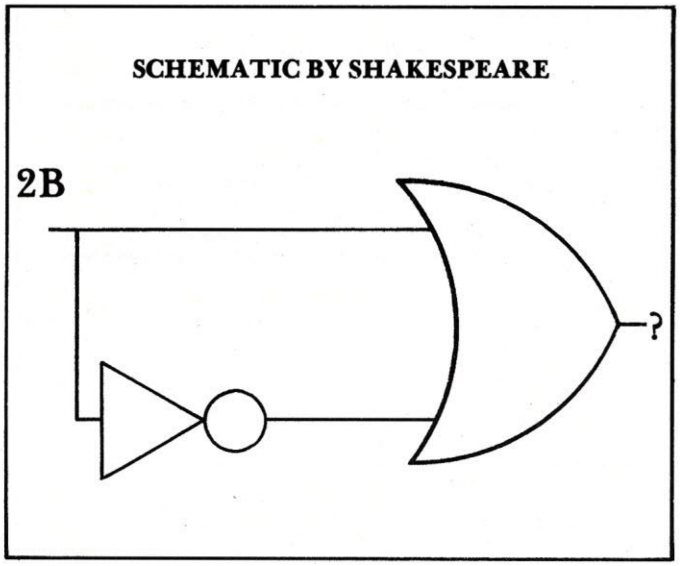
I will discuss these topics in future issues in terms of the mini- and microcomputer. This column is intended to stimulate an exchange of ideas relating to the implementation of artificial intelligence on the typical home computer. Hopefully, you will participate in the future content of the column; let me know what AI problems interest you and I will try to cover any such topics which seem to be of general interest to the microprocessor and the home hobbyist community.

In future discussions of AI areas such as game playing by computer, I will try to find solutions which you can implement on your microprocessor. (Solutions for many AI problems are difficult because they tend to be both memory and processor intensive.) If a solution in BASIC or some other microprocessor language is not readily available, I will attempt to describe the solution in terms that will help you to implement such a solution or partial solution on your own system. Since BASIC has become the programming language most commonly used on microprocessing systems, I will discuss the problem and its solution in terms of programming the solution in BASIC.

Remember, too, that your programming solutions to the topics covered in this column are welcome. Please send a complete and documented source code as well as sample runs and user instructions along with any program you submit. Interesting problem solutions will be published in future columns. And if you want additional information on any of the topics, I will gladly send you a list of works on the subject.

From time to time, I will be giving you my opinion on topics in AI and how they relate to both the computing community and society in general. It is only fair, then, that I should also include your opinions on topics relating to artificial intelligence and its impact on society. Ideally, this column will become an open forum for opinions relating to all aspects of artificial intelligence.

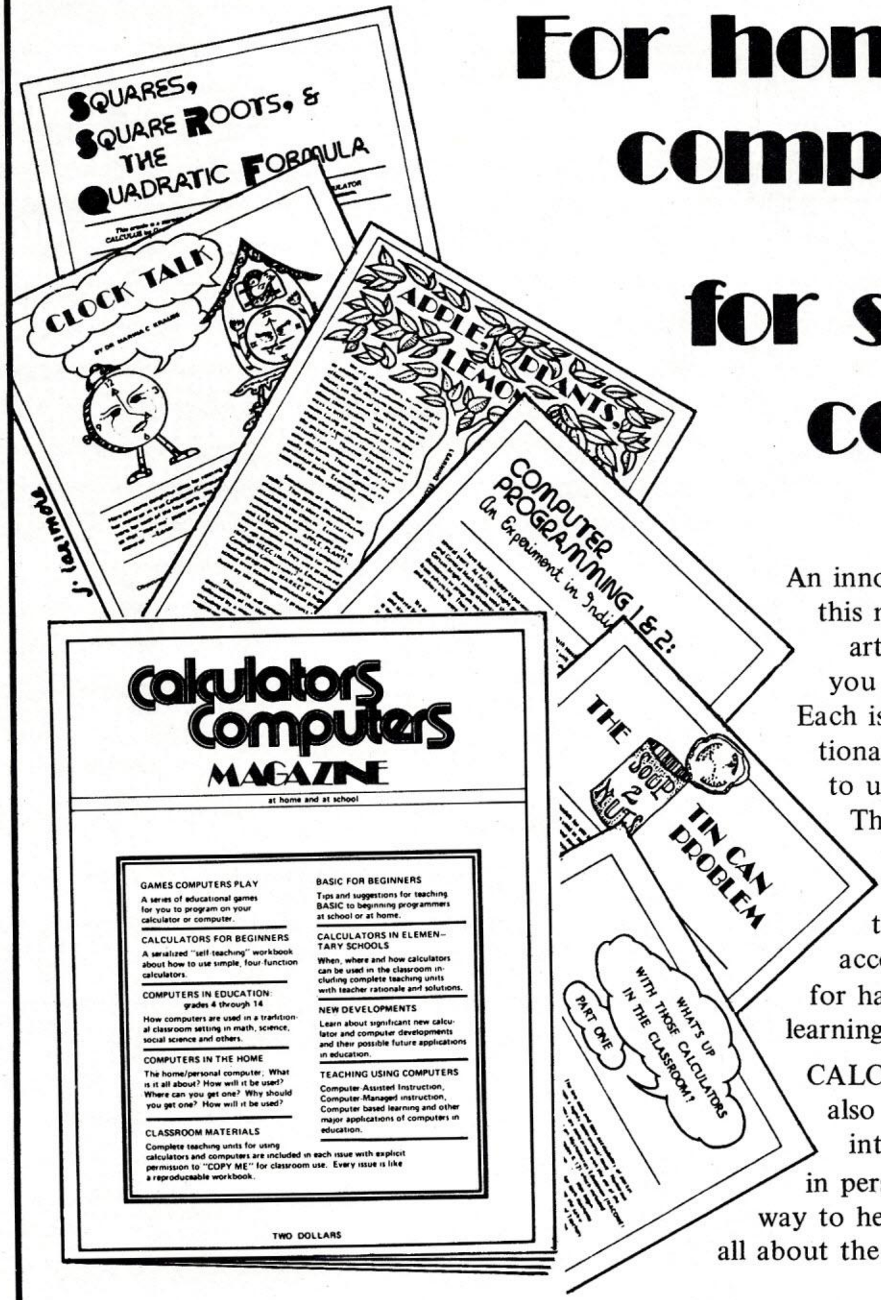
It is obvious that artificial intelligence is an extremely broad area. Researchers from the fields of psychology, linguistics, computer science, and education, just to mention a few, are spending a great deal of time and effort in solving many of the subproblems associated with the ultimate goal of understanding and duplicating human intelligence in computers. I hope that this column can bring a better understanding of some of the most interesting AI problems to the fastest growing area of computer science—the home computer hobbyist. ▼



Cartoon by Andrew Singer

For home computing...

for school computing


SQUARES, SQUARE ROOTS, & THE QUADRATIC FORMULA
CLOCK TALK
APPLES, PLANTS, LEMONS
COMPUTER PROGRAMMING 1 & 2: An Experiment in Indi
THE SOUP 2 NUTS TIN CAN PROBLEM
WHAT'S UP WITH THOSE CALCULATORS IN THE CLASSROOM?
PART ONE
J. Larimore
Calculators Computers MAGAZINE
at home and at school

GAMES COMPUTERS PLAY
A series of educational games for you to program on your calculator or computer.

CALCULATORS FOR BEGINNERS
A serialized "self-teaching" workbook about how to use simple, four-function calculators.

COMPUTERS IN EDUCATION: grades 4 through 14
How computers are used in a traditional classroom setting in math, science, social science and others.

COMPUTERS IN THE HOME
The home/personal computer; What is it all about? How will it be used? Where can you get one? Why should you get one? How will it be used?

CLASSROOM MATERIALS
Complete teaching units for using calculators and computers are included in each issue with explicit permission to "COPY ME" for classroom use. Every issue is like a reproducable workbook.

BASIC FOR BEGINNERS
Tips and suggestions for teaching BASIC to beginning programmers at school or at home.

CALCULATORS IN ELEMENTARY SCHOOLS
When, where and how calculators can be used in the classroom including complete teaching units with teacher rationale and solutions.

NEW DEVELOPMENTS
Learn about significant new calculator and computer developments and their possible future applications in education.

TEACHING USING COMPUTERS
Computer-Assisted Instruction, Computer-Managed instruction, Computer based learning and other major applications of computers in education.

TWO DOLLARS

An innovative approach to computing, this new magazine contains creative articles and materials that will teach you all about calculators and computers. Each issue includes self-contained instructional units which can be reproduced to use at home or for classroom use. These "copy me" materials range from elementary school level through community college. A teacher's commentary or guide will accompany each unit. Some units are for hand-held calculators, others for learning BASIC.

CALCULATORS/COMPUTERS Magazine also includes games, simulations, and interesting articles for anyone interested in personal computing. Here is an exciting way to help your family and friends learn all all about the joys of computing!

ORDER YOUR SUBSCRIPTION NOW!

Calculators/Computers Magazine - 1 yr. (7 issues) $12.00

Payment MUST accompany all orders.
U.S. currency only please.

☐ check enclosed
☐ Mastercharge Card #____________________
☐ BankAmericard Exp. date____________________

NAME____

ADDRESS____

CITY__________________STATE__________ZIP_____

Foreign Rates: $17 per year surface mail. Airmail to Canada $23; Airmail to Europe and Pan America $28; Airmail elsewhere $32.

Subscriptions begin with the current issue. Published October, November, January, February, March, April, May.

Send order and payment to:

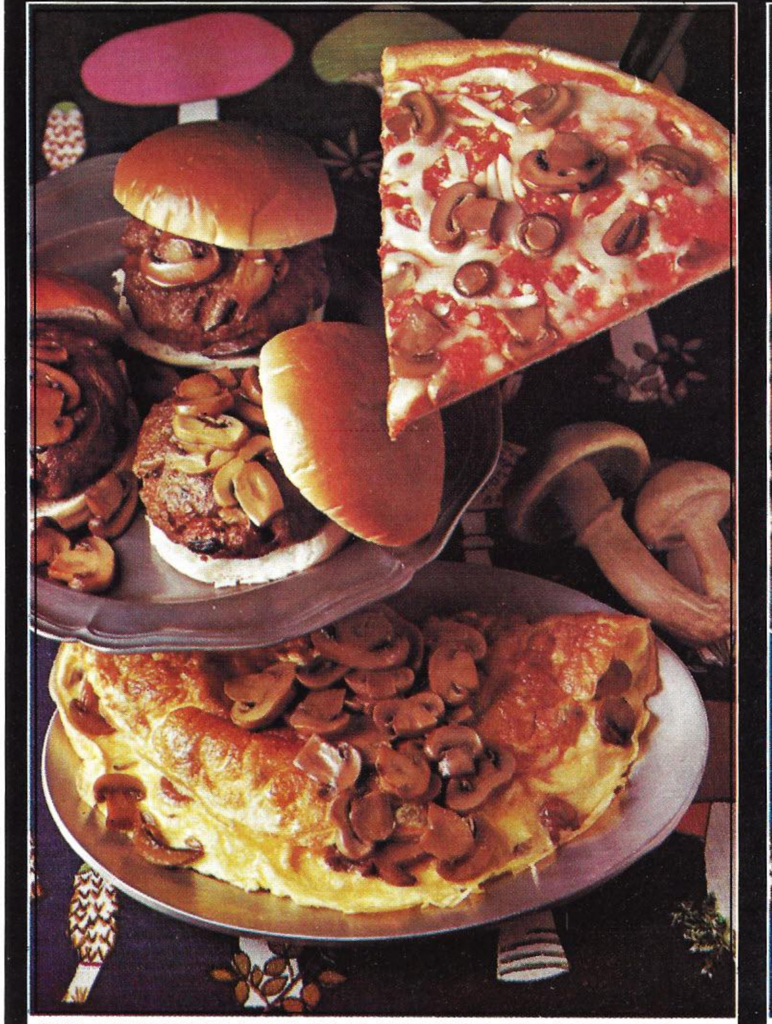
DYMAX
P.O. Box 310
Dept. 25
Menlo Park, CA 94025

COLLIER IS THE BEST ENGRAVER, PRINTER IN THE COUNTRY.

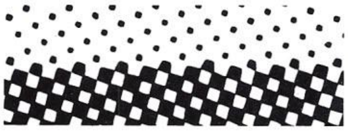
Because. A revolution in engraving has happened. Collier split the dot. And because of the new Laser Beam, Collier is now light years ahead of the rest. The laser does it. Here's how it works. Technically, the laser beam which is used to control film exposure (thus "Program" the engraving) is split into six sectional beams. Each of these is digitally modulated by computer to transfer half-dots of picture information per scanner revolution. Since two picture-information "bits" are available per dot in circumferential direction, it's possible to expose a smaller area in the second "orbit"...or even completely omit it (thus producing an actual half-dot or even an elliptical mini-dot). If that seems to all sound a lot like gobbledygook, don't worry about it. The important thing is, it's here and it does work and it gives your image resolution that you never could get before. We'll be happy to demonstrate it for you. Even happier to produce your next set of four-color letterpress, offset and gravure engravings. Call Collier Graphic Service Company Inc., 240 West 40th Street, New York, New York 10018/(212) 840-0440.

ATLANTA	BOSTON	DETROIT	HARTFORD	NEW YORK	PHILADELPHIA
(404) 892-2383	(617) 965-5660	(313) 259-2111	(203) 367-0706	(212) 840-0440	(215) 988-0110

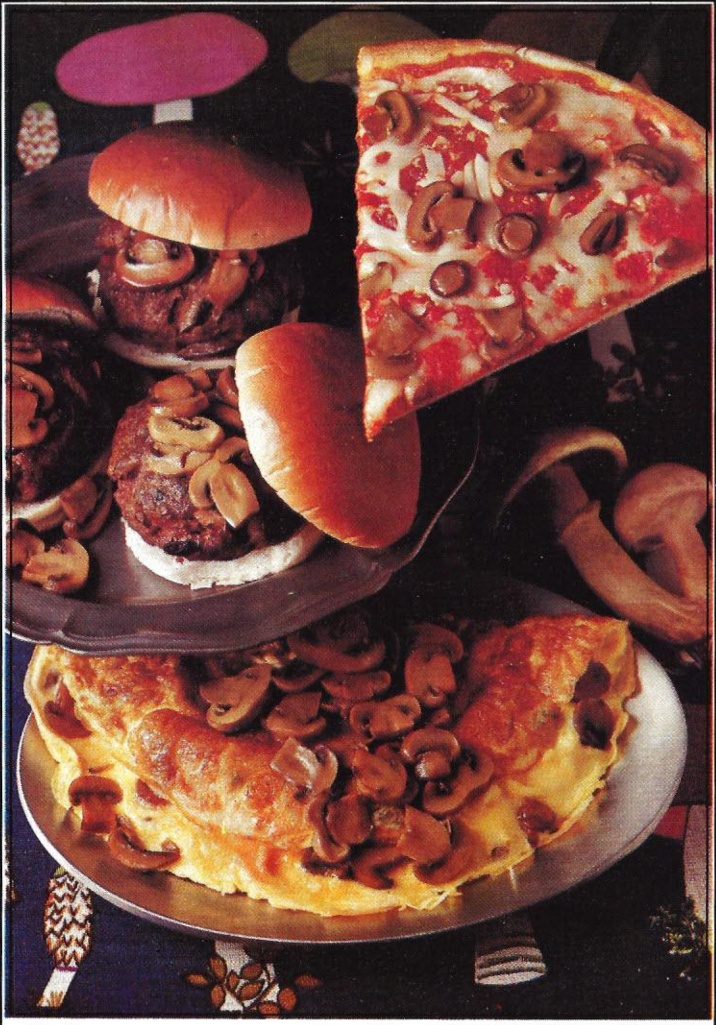
OR CALL TOLL FREE (800) 221-2585/(800) 221-2586



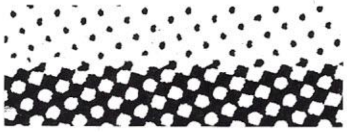
THE ABOVE ENGRAVING WAS MADE WITH THE CONVENTIONAL PLATE MAKING PROCEDURES, AS YOU CAN SEE, IT LACKS COLOR FIDELITY AND SHARPNESS, IF YOU COMPARE IT WITH...



THE CONVENTIONAL ENGRAVING DOT.

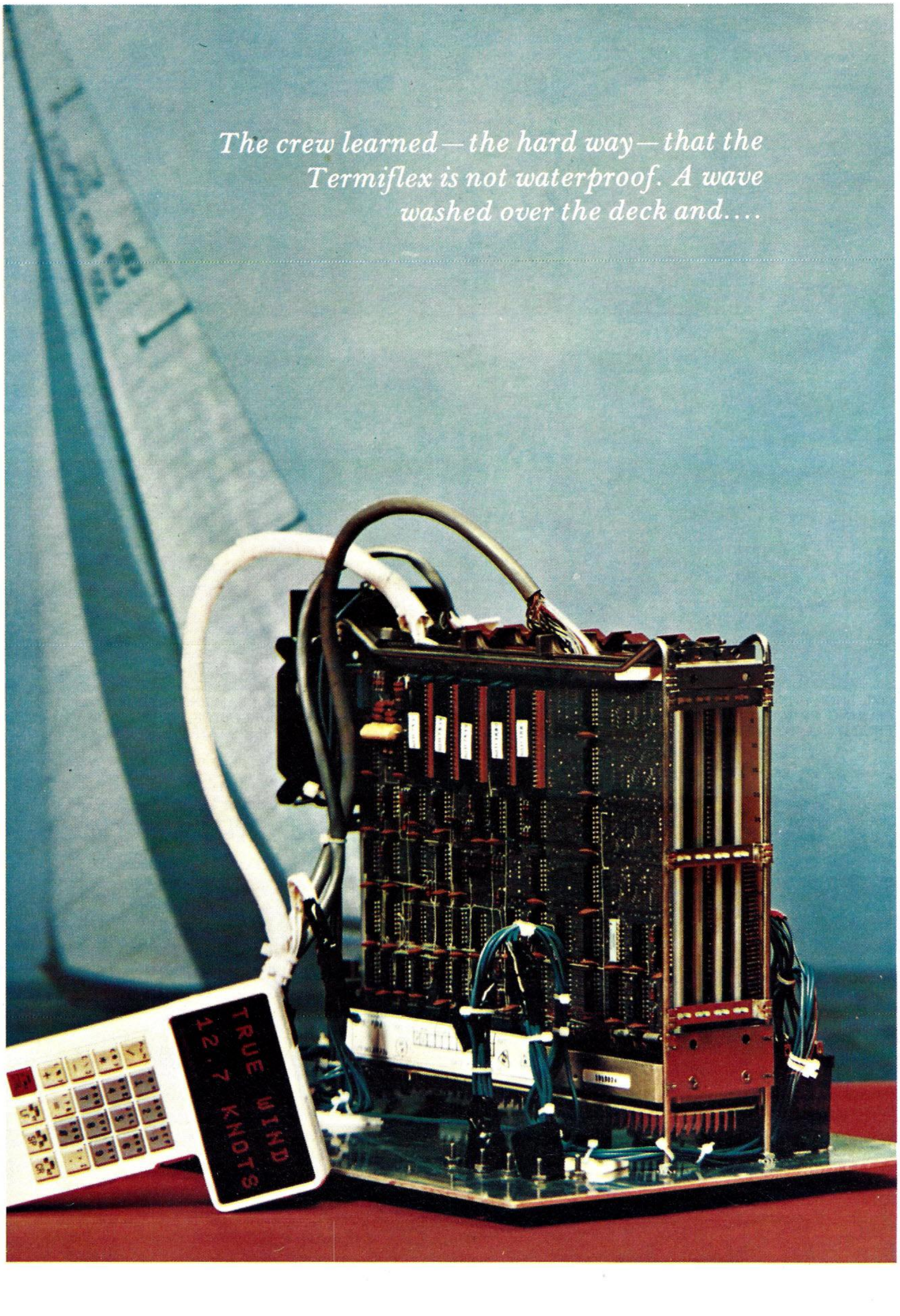


THE COLLIER'S LASER BEAM ENGRAVING, AS YOU CAN SEE FROM THE ABOVE, HAS BRILLIANCE, SHARPNESS AND COLOR FIDELITY THAT ONLY COLLIER'S DOT CAN PROVIDE.



THE COLLIER LASER DOT.

THE COLLIER ADVANTAGE. COLLIER GRAPHIC SERVICES COMPANY, INC., 240 WEST 40TH STREET, NEW YORK, NEW YORK 10018


The crew learned — the hard way — that the Termiflex is not waterproof. A wave washed over the deck and....

TRUE WIND
12.7 KNOTS

ROM
COMPUTER APPLICATIONS FOR LIVING

COMPUTERS CHALLENGE AMERICA'S CUP

by Eben Ostby

It's not enough that computers are running the scoreboards in stadia across the country; not even enough that they help analyze baseball scores and football strategy and give summaries of wrestlers' moves. They've now gotten into one province that seems totally alien to the rigid, mathematical world of the computer: personal computers are now an important part of the fluky wet world of sailing.

Even those of us who don't object on aesthetic grounds may find it hard to believe that the little, fragile electronic computer can prove useful aboard ship. The rigors of sailing are easy to underestimate (computers can't wear foul-weather gear), and most of sailing seems like an art rather than a science; it certainly takes an artist to be able to balance currents, winds, the boat's performance on different headings, the weather, and the crew. Sailors aboard the America's Cup defenders *Courageous* and *Independence*, however, have found a personal computer to be invaluable: it made the difference which probably gained the Cup.

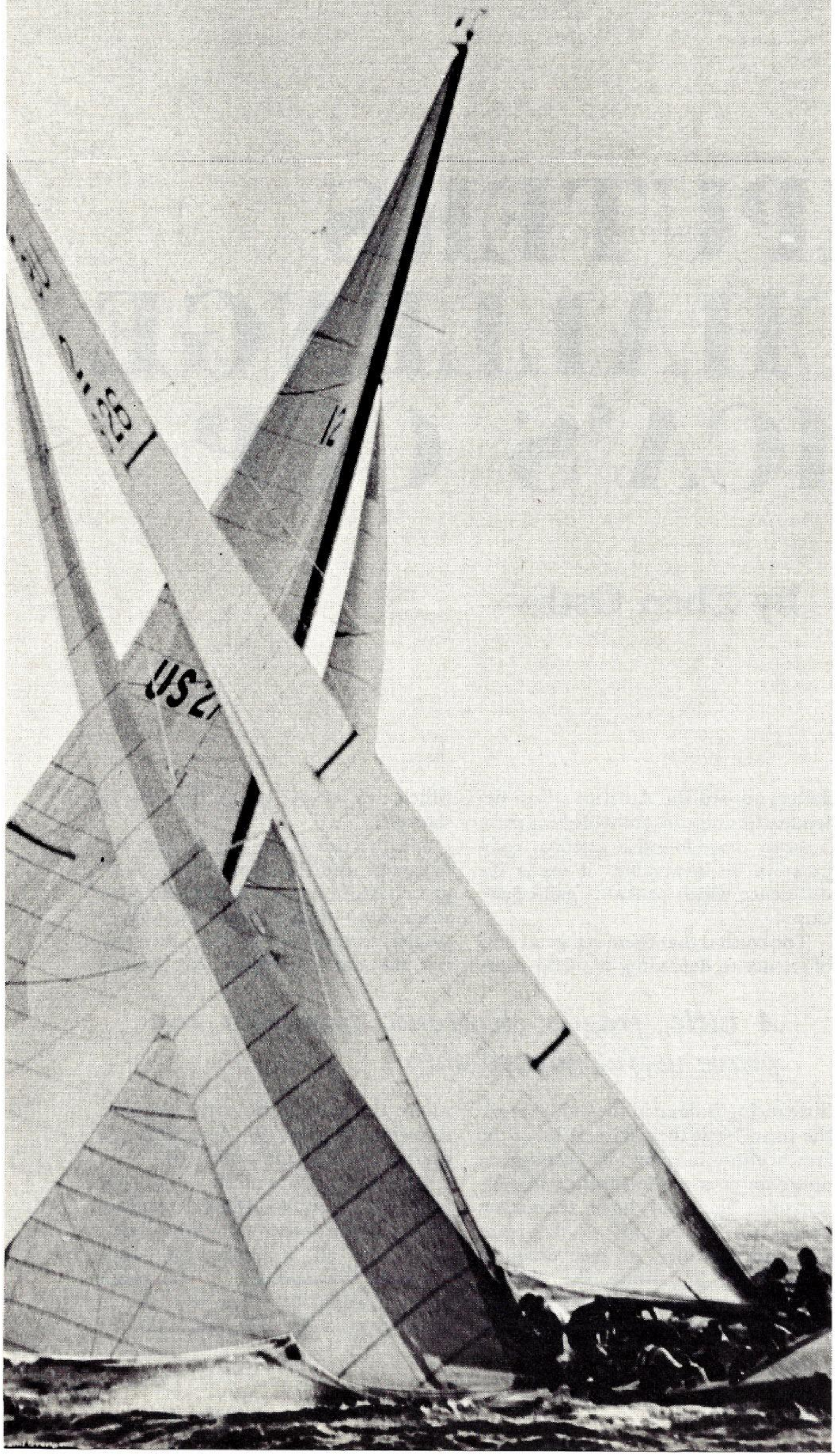
The truth is that there is a good deal of science to defending the Cup—very little racing is done in the old "seat-of-the-pants" style these days. A lot of the crew's effort is spent on keeping an optimum course, the absolute best attainable. Since the boats themselves have been "tuned up" to the point where improvement on their design is pretty much impossible, the winning boat will most likely not be the one which has the best design, but the one which can cut the course down to its shortest.

> *A little, fragile electronic computer can prove useful aboard ship.*

Ideally, this is the job of both the navigator and the skipper. The navigator has to keep track of a myriad of instruments—wind gauges, water-speed gauges, compasses, and so on—and use the data to figure out exactly where the boat is at each moment. Knowing this, he can determine the shortest route to the next mark.

But if there's a little error—for instance, if his wind-speed readings were not taken at representative times, and were all a little high—then the navigator's picture of the boat's location would be a few feet off. In these races, a few feet can make the

Photographs courtesy of Digital Equipment Corp.



difference between winning and losing. That's why the navigator of the *Independence,* Pete Lawson, saw the need for a personal-computer-based navigation system.

The system he envisioned would let the navigator input all the data he had available and would instantly perform the necessary computations to determine the boat's course and position. This would save enough time to make more calculations possible. Having it all computerized would help eliminate computational mistakes, too, and would increase the accuracy of the computations.

The first system that began to fill the bill was a programmable calculator with programs on magnetic strips to do the important calculations. As good as a calculator was, however, it needed improvement: to change programs frequently was inconvenient, and the results of the calculations were limited to numbers that could be displayed on the calculator's LED display. Luckily, however, one day navigator Lawson met an engineer from Digital Equipment Corporation in a bar in Newport; it happened that Digital's microcomputer, the LSI-11, was ideally suited to the task of assistant navigator for the Cup defenders.

Another engineer from Digital, David Schanin, solved the first problem: the simple but crucial one of how to communicate with the processor. Unlike a calculator, the LSI-11 is just a computer—it has no switches or lights on the front panel. It's just a couple of circuit boards in card-guide. So if you want to use it for anything, you have to find an appropriate set of input-output devices.

Usually someone designing a system will choose a standard terminal for input and output; a Teletype is very common. Well, that would be nice to use on board...but have you ever tried to sail a boat with a Teletype that has to be kept dry and stable? Salt spray could do wonders to the innards, and the paper would get all soggy. The biggest problem with a conventional terminal is that it's generally pretty big, and there's not a lot of space for a terminal on board a twelve-meter yacht. So David and his associates chose an esoteric little terminal called the Termiflex.

The Termiflex rates an article in its own right. What makes it unique is that it is probably the only truly hand-held terminal you can buy. It looks somewhat like a pocket calculator, though it's a little bit bigger than most. At the top is a two-row LED display, like that of a calculator, except that it can display any ASCII character. All of the rest of the face is taken up by a keypad that's also like a calculator's—except that each key has four characters instead of just one inscribed on the face. The largest of these figures are the standard numerics and arithmetic symbols, so that at first glance the keypad does look like a calculator's. But the other three inscriptions on each key are of standard alphabetic characters. In a way, the result reminds one of a Touch-Tone phone.

By pressing one of three buttons on the side of the Termiflex, the proper letter on the keypad is selected. So, although the terminal takes two hands to run, it is capable of sending any normal characters to the computer. And it can display twenty characters at a time. Since it is small and light, it was perfect for use on board the Cup defender.

Well, almost perfect. The crew learned—the hard way—that the Termiflex is not waterproof. It happened when a wave washed over the deck and...well, you can guess what happened to the Termiflex. But, after the engineers fashioned a little waterproof Dacron bag for it, the terminal proved ideal.

The computer itself, the DEC LSI-11, likewise had to be protected from the elements. Leaving the exposed rack as it came wouldn't do, since salt air corrodes anything it contacts, and damp salt air is worse. The very real possibility of water damage—even if the computer was mounted out of the way, below decks—also had to be considered.

The system was a triumph—for the computer designers as well as the sailors.

The obvious solution was to seal the computer in a box that was air-tight. The less obvious problem was that computers, even tiny ones, generate heat. Normally a fan can blow enough cool air by the computer to keep it from burning up. But in sealing off the outside (damp) air, the engineers also sealed off the supply of cooling air.

How did they handle this situation? Simple: they made the computer's box big enough—three feet high—to allow surface area enough to radiate all the heat the LSI-11 produced. Then they put a little fan *inside* the box, to make sure the air inside would circulate and allow the box to radiate the heat.

The resulting box still got up to 120 degrees at times—the limit of the LSI's operating range—but the computer never once broke down. Naturally, a breakdown would have been catastrophic, since a backup system wasn't available on board. And the most damaging part of running at high heat is temperature change: the rapid increase in temperature that occurs when the computer is turned on in the morning. Despite a daily increase in temperature from 60 to 100 degrees, the system continued to perform.

Have you ever tried to sail a boat with a Teletype that has to be kept dry and stable?

In order to keep the system running without a failure, the engineers had to go to considerable lengths. According to engineer David Schanin, "The system was incredibly over-engineered. That's what we had to do in order to get something close to one hundred percent reliability." In places where one wire would have normally sufficed, they put in three to five wires. Where most people would have used one sixteen-gauge wire, they used two fourteen-gauge wires. They also used heavy aluminum for the box in which the computer was sealed.

The power supply was engineered in the same fashion. Although the system drew at most about eight amps, the power supply used twenty-five-amp regulators—just in case. The components used were especially good (and expensive) ones. Despite this, there were still problems with the power supply at first. ("I lost a lot of sleep over that power supply," said engineer Schanin.) After replacing some faulty parts, however, there were no further difficulties.

The initial version of the system included just these components: the computer, the input-output device, and the power supply. There was also a paper-tape reader for loading the programs into the computer each morning; since the computer has a dynamic memory, the programs were wiped out when power was shut off. Because it was only used once a day, this reader was not stored on the boat itself. Instead, it was attached to the boat in the morning, the tapes were read in, and then it was disconnected and put away for the day.

The only real advantages the computer had over a programmable calculator at this stage were that it was more flexible in its I/O; it could hold more programs, and the programs could be written in a higher-level language; and it ran on big, dependable Die-Hard batteries—car batteries—rather than penlight batteries, which tend to give out at inopportune times.

Realizing that a computer *can* do much more than that, the designers increased the scope of the system to make it collect its own data and report on it, automatically. This change turned the system into a special-purpose navigation system, with power now far beyond that of a calculator.

The Cup defenders had a number of digital instruments on board already. These included an anemometer (wind-speed indicator), a digital wind-direction indicator, and a boat-velocity meter. These devices, made by Signet, collected the data from sensors located around the boat, and averaged the readings in a buffer. An LED display reported the data to the crew.

What the computer engineers did was tap in to the meters after the buffering stage. This was important, since a meter's buffer served the function of an averaging device. The averaged signal was updated by the meter at regular intervals. The computer, properly tied in to the meter, could read the value of the instrument just by sampling the appropriate input line. This was done by means of a standard interface called a *parallel line unit*. The parallel line unit served the function of putting the input signals into a standard form for the computer's use.

Another very important navigational instrument on board both *Independence* and *Courageous* was a digital compass. This was a device to determine the sine and the cosine of the angle of the boat relative to magnetic north. The computer sampled this input every 100 milliseconds. Then a program would convert this data to the boat's actual heading. This information coupled with the current time of day (supplied by an accurate crystal clock) enabled the computer itself to check all the instruments it needed to determine its course.

For instance, if the boat was sailing with the wind abeam (coming across the side of the boat), the computer had to take into account the wind direction, the speed of the boat, the amount the boat would "slide" relative to a straight course, the current, the wind speed, etc. This calculation was done every half second—much more frequently than a person could have done it.

Other information was computed by the system either on demand or periodically. The boat's actual velocity—not relative to the moving water, but relative to the stationary land—was very important. The effects of currents were computed from information about the local tides. The program even accounted for cyclic variations in the wind. The sailors found that the winds were not constant; that they would increase and decrease in intensity at regular intervals. The computer could easily monitor the wind patterns, and therefore predict the wind with some measure of accuracy.

By taking into account the speed of the boat, the computer could compute the true wind direction. (Since the motion of the boat affects the wind vane's reading, the program has to subtract out the effect of the moving boat.) Similarly, the programs compute "speed made good," which is the progress that the boat is making toward the next mark, regardless of how it has to head in order to tack upwind.

The computer also worked with statistics on the boat's performance. By monitoring the length of time it takes for the boat to tack, it is possible to rate the crew's performance in tacking. They can then note anything that gives them even a tiny gain—the minute gains are the little things that add up to a victory which is measured in seconds, not minutes.

Sailing is, of course, a skill that tries to balance the conditions of the boat and the course. The captain who can bring his boat closest to the optimum will likely win the race. The captain must tread the thin line between heading up too far into the wind, and thus losing speed, and heading off (downwind) too far, and thus aiming away from the next racing buoy. This skill demands not only a sailor's knowledge of his boat, but also the precise direction that his course should take him. A computer can keep track of the proper course and compare it to the boat's actual position—in that way, the captain has continuous feedback on the performance of his boat and crew. He is better able to make the proper decisions.

Of course, adjusting to a personal computer as a tool may be a problem in itself. Despite the complexity and power of the computer used by the Cup defenders, it had a simple, clear method of accessing the data. All data collection was done automatically so no person had to intervene to get the thing to work. Then, output was produced at the touch of a button: one keystroke provided enough information to the computer to allow it to decide what function to perform. And in case the user had punched the wrong button, the display identified the output and the mistake was obvious. For instance, the calculation of the actual wind speed (not the speed relative to the boat's travel) was displayed not simply as "12.7" but as "TRUE WIND 12.7 KNOTS" on the Termiflex.

However, as such things will happen, there were problems getting people used to the idea that a computer can help navigate a boat. Crew members were reluctant, at first, to use it. Doubts were dispelled, however, according to engineer Schanin, when they were forced to sail without the system—and their performance suffered.

The performance of the computer system itself was easily improved when necessary. Since it was, deep down inside, really a Digital PDP-11 computer, it was possible to use DEC's FOCAL high-level language. FOCAL is an interpretive language, but one that lent itself well to the kind of flexible I/O this application demanded. Since it was an interpreter, it wasn't necessary to recompile the source every time some change was implemented. In fact, one reason it was possible to complete the engineering task so quickly (about two man-months of work were involved) was precisely because a high-level language was used instead of assembly language.

In order to use FOCAL, though, a few modifications had to be made to the language processor. In particular, the digital compass required that the data be accessed within 100 milliseconds of when the compass interrupted the program. Since FOCAL takes a while checking out program interrupts—in this case, over 100 milliseconds—the designers had to write a new interrupt handler for FOCAL. But there was little else that had to be done to the software. In fact, some of the routines that were used were basically translations of routines that had been used previously on the programmable calculator.

The resulting system was a triumph, both for the computer designers, who saw that the system could determine accurately the boat's position to within fifty feet after a six and a half mile race, and for the sailors, who, after all, successfully defended the Cup. And if that's not the ultimate test of a computer system...! ▼

A BEGINNER'S GUIDE TO PERIPHERALS

INPUT/OUTPUT DEVICES YOUR MOTHER NEVER TOLD YOU ABOUT

by Leslie Solomon and Stanley Veit

From *Getting Involved With Your Computer,*
Copyright © 1977 by Ridley Enslow Publishers.

A digital computer is designed to do things, but it's just going to sit there unless you can somehow get data from the outside world into the computer and out again after processing. That is where peripherals and I/O (input/output) ports enter the picture.

The data in the computer is generated on the eight data lines, assuming an eight-bit computer, but unfortunately all eight bits are present on the data lines at one time. (This is called parallel data.) This condition is at odds with many I/O devices such as the Teletype or video terminals that operate with serial data (one bit at a time). Therefore some means must be found to perform the necessary two-way conversion.

The most common way to accomplish the two-way serial-to-parallel and parallel-to-serial conversions is with a device called a UART, short for Universal Asynchronous Receiver Transmitter. Essentially, this mouthful describes an IC that accepts digital data in parallel form and transmits the data in serial form. The other half of the IC receives the data from the outside world in serial form and waits until it has the complete eight-bit byte. It then dumps the complete byte in parallel form onto the computer data bus. This explains the meaning of receiver and transmitter, but what about *asynchronous?*

In data communications there are two basic types of serial transmission. One is called synchronous because the incoming data words are locked into the computer timing. If the timing between the

ROMtutorial ROMtutorial

IC (integrated circuit): A transistor (not a radio) is a tiny speck of specially-processed silicon having three different layers. Integrated circuits contain hundreds or thousands of transistors formed in the same block of silicon, along with the necessary interconnections. The block is still quite small, so it's called a chip, and is often packaged in an inch-long block of plastic with metal connecting pins coming out. Often this package itself is called the chip.

Handshaking: Control signals that allow two electronic circuits to synchronize their operations. A computer sends out some predetermined signal—a tone, a bit sequence, or a character. Then the thing it is trying to communicate with sends back a response—a different tone, another sequence of bits, or a special character. When the computer receives this signal, it "knows" that the connection is complete so that communication can start.

CRT: Cathode ray tube. An electronic vacuum tube containing a phosphor-coated screen on which information may be displayed by the use of controlled beams of electrons; the electrons strike the screen coating, causing it to emit light. Television sets use them.

Hz (hertz): A unit of frequency equal to one cycle per second.

Baud-rate strapping: In order to set the baud rate for the UART, it's necessary actually to modify the circuit board so the right wires connect up. Usually a number of circuits are included on one board, and the right one is selected by attaching a little wire, or strap, to connect the appropriate clock with the rest of the circuit. On some devices, a switch is included instead, in order to do the same thing.

Filling a buffer: A buffer is a temporary storage area which is used to equalize or balance different operating speeds. For example, a buffer can be used between a slow input device such as a typewriter and the main computer, which operates at a very high speed. When it is used to collect characters from an I/O device, the program that uses that data can look at an entire string of characters without having to read in each individual one. In this process a separate program reads data into the buffer until it can hold no more characters. If the buffer is eighty characters big, when the eightieth character is in, the buffer is filled. The program returns control to the calling program, which does something with or to the eighty characters.

computer and the serial device are not in step, garbled transmission will result. Because most computers operate rapidly and peripherals very slowly [Teletypes operate at only 110 baud (bits per second) and video terminals operate up to 9600 baud], some means must be found to get the data from the outside world in step with the computer. This is where *asynchronous* comes in.

The beginning and end of each word can be identified by designated start and stop bits. Now, as long as both ends of the system (the computer and the external device) can agree on the baud rate (usually determined by the setting of the I/O board), the start and stop bits that are added to the digital word can tell the computer when a word starts and when it ends. There can be any length of time between words as long as the start and stop bits are added to do the *handshaking.*

The 110-baud rate mentioned for the Teletype is that machine's standard speed both for the keyboard and tape reader. It means that 100 words per minute are being transmitted. This is the industry standard for slow-speed data exchange. This code takes 100 milliseconds to send one character. The 300-baud rate (generally used by CRT terminals and the hobbyist's tape system) means that 300 words per minute can be transmitted. Although at first this may seem fast, keep in mind that a typical computer can operate up to 25,000 baud!

Now, let us take a look at the operation of a UART (see Figure 1). Although our discussion relates to the UART, some manufacturers are supplying other UART-like devices that do the same thing but under software control.

A UART needs a clock—something to keep all the data bits in step. Because of circuit considerations, a UART requires a clock frequency that is sixteen times the selected baud rate; for example, a 110-baud rate circuit must have a clock with a frequency of 1760 Hz. For the 300-baud rate the clock must operate at 4800 Hz. Most I/O boards have some form of *strapping* to set the baud rate (actually they are setting up the clock frequency to the correct speed). Baud-rate strapping is usually explained in the manual for the I/O board.

In the UART transmitter section the computer "fills" a buffer with the required number of bits and after a *handshake* signal (which occurs when the buffer is filled) allows the data to exit one bit at a time. Each data bit must be in step with a clock pulse. In the UART *receive mode* the outside device fills the buffer, and when the buffer is full a handshake dumps the complete digital word into the computer in parallel form.

Now let us follow the UART-transmitted signal which has assumed serial form and see how it gets to the TTY or CRT terminal.

Before this can be done we must have some understanding of how a

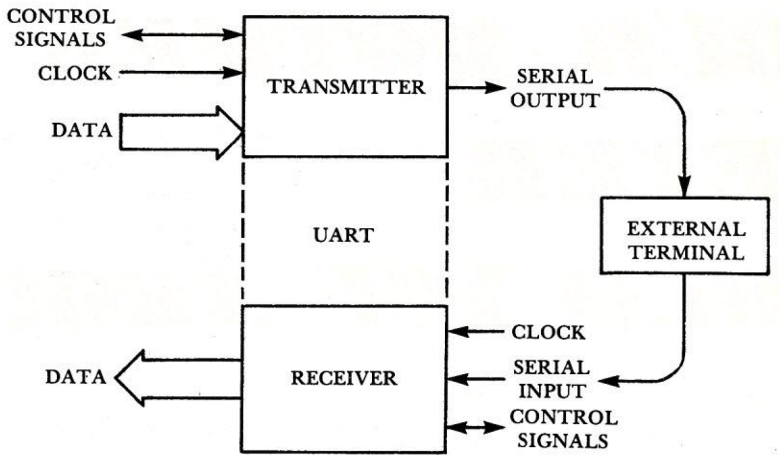


Figure 1

A UART contains a logic receiver and a logic transmitter. It allows communication with the outside world.

Teletype works. A TTY contains two sections: a transmitter consisting of the keyboard and any associated tape reader and the receiver consisting of the printer and any associated tape perforator. Although they are two separate sections, they are mounted in the same housing and appear to be one piece. They are not electrically connected unless the machine is in *local* mode (via a front-panel switch) when the keyboard signals are passed directly to the printer. In local mode the machine communicates only with its printer, not with the outside world.

The other front-panel switch position is called line and it allows the two halves of the TTY to communicate with the outside signal source. Four wire leads are required; two for the printer and two for the keyboard.

The keyboard (transmitter) section of the TTY uses special electromechanical circuits that convert each keystroke into the pertinent ASCII code character. What is ASCII? The name ASCII stands for American Standard Code for Information Interchange and consists of various combinations of 1s and 0s to form all alphanumeric characters, plus a wide variety of control characters. ASCII is a special language that computers and I/O devices use to communicate with one another.

Looking at Figure 2, let us see how data are passed back and forth between the computer and the TTY. When the computer sends data to the TTY, it makes and breaks a special 20-mA current circuit (usually on the I/O board). This current circuit drives special electromechanical devices in the TTY serial-to-parallel converter. Once a complete *word* is accumulated, the mechanism causes the pertinent TTY key to strike the paper and make its mark. This flow is d.c. and therefore has a positive and negative lead. The TTY input connectors are appropriately marked so that the correct connections can be made. If the leads are reversed, the TTY will not print. In some Teletypes a tape perforator (a mechanical punch that makes hole patterns in paper tape 1/2 inch wide) can be used instead of the printing mechanism. This system is employed when a program is to be saved on paper tape.

When a key on the keyboard is activated, it operates an electromechanical system that makes and breaks a switch with the ASCII code for the letter on the key. In the input section of the I/O port a dual-polarity voltage source is keyed by the switch closings from the TTY. A positive voltage at the port causes a 1 to be entered and a negative voltage produces a 0. Because the TTY in this case merely opens and closes a switch, this portion of the TTY is not polarity-sensitive and the two leads for this section can be connected in either way on the two input connectors.

Therefore a Teletype requires two kinds of signal source: a current source for the printer mechanism and a voltage source for the keyboard output.

ROMtutorial ROMtutorial

Receive mode: The mode or state in which a device can accept data from an external source. Some terminals, for instance, cannot receive and transmit simultaneously. They start in transmit mode, sending characters until a RETURN is pressed. Then they go into receive mode and are unable to send characters until they receive a signal from the computer saying that it has finished sending information.

Local mode: A state in which a terminal or another device does not communicate with the computer. In local mode a terminal acts like a normal typewriter.

Alphanumeric: Information made up of sequences of characters containing letters (alphabetics) and digits (numerics).

20mA current: An ampere is a unit of electrical current—about one ampere flows through a 100-watt light bulb. 20mA is 0.02 of one ampere; it is a very small current by light-bulb standards. Computers, however, usually have currents in the milliampere range trickling through parts of the processor, so 20mA is not so small in that context.

Word: An ordered set of characters that is treated by the computer as a single unit; for example, eight bits make one byte, usually called a word. Words, however, can be as long as seventy-two bits long.

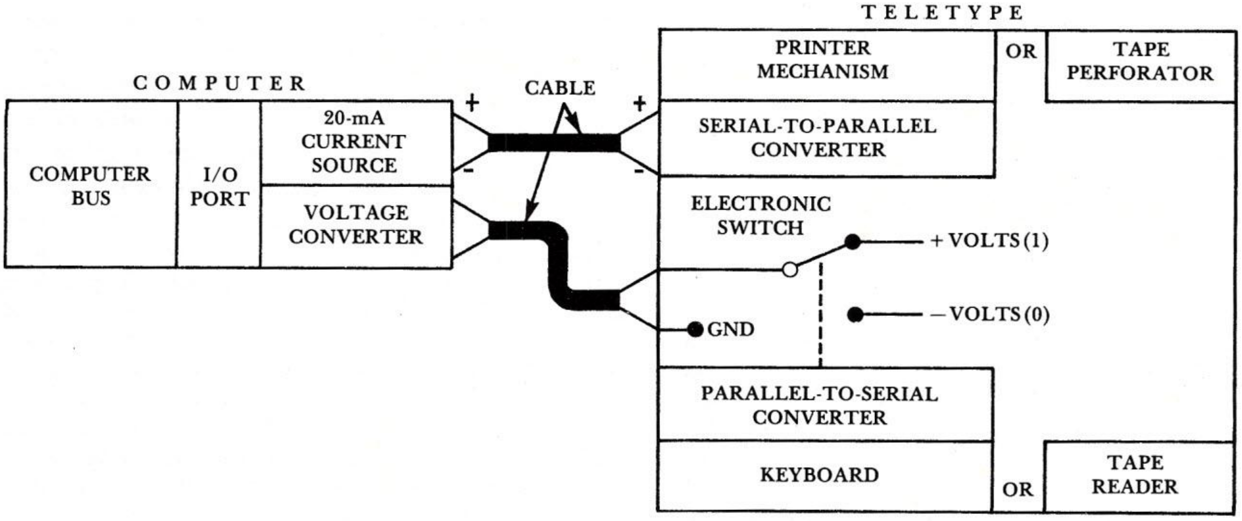


Figure 2
How data passes between a computer and a Teletype.

The video terminal on the other hand uses the same voltage scheme (positive for 1 and negative for 0 for both sections—keyboard and CRT readout). This system is usually referred to as the RS-232.

There are other device communication standards, but at this time the hobbyist is concerned only with the 20-mA current loop and the RS-232 standards.

Parallel I/O Ports

Parallel I/O ports is a new term which means that the computer data bus eight-bits (byte), plus the handshake signal, are all available at the same time; then why not dump all the pertinent data at once? Special circuits called, interestingly enough, parallel ports, can be connected to the computer if high-speed data I/O is desired. By using this approach it is possible to insert, say, 8K BASIC into a computer in a matter of seconds rather than in the ten to twenty minutes needed by the slower 110-baud TTY approach. One simple example of a useful parallel-port entry device is the Oliver tape reader which is available at most computer stores. More on it later.

Mass Storage—Cassette

Up to this time we have talked about printing data on a TTY or displaying the data on a CRT screen. Now, what can you do if you want to save a program or some data? Of course, you can always perforate a paper tape by using the TTY, but such TTYs are expensive and the process is slow. You can always observe the data on the CRT and write it down on a piece of paper and then painfully type it into the computer whenever you need it. If you have lots of money, you can use a floppy disk (more on this later), but that method is expensive also. There must be an easier way!

Some disks can store millions of digital words.

The easy approach is called cassette recording which uses a conventional low-cost, audio-cassette/tape recorder/ and a low-cost cassette. But, wait a moment, you may say—a digital computer operates with discrete digital 1s and 0s and a tape recorder operates with speech or music. How is this done?

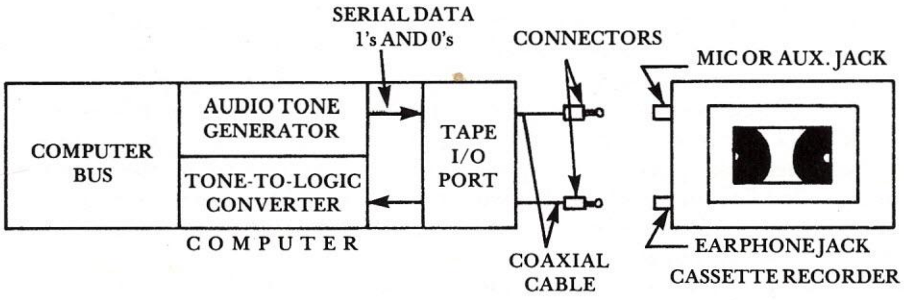


Figure 3
Communication between a computer and a cassette recorder.

Suppose we have selected two different audio tones, both within the recording range of the cassette, and use one tone for 1 and the other for 0? In this way we can use the low-cost cassette recorder and its cassettes.

Although there are many different tone standards, and just about every computer manufacturer has his favorite, some use what is called the "Kansas City standard," which was set up by a group of hobby-computer manufacturers back in 1975. The Kansas City standard uses 2400-Hz for 1 and 1200-Hz for 0.

During the last year or two many peripheral manufacturers introduced variations of interface boards for cassette recording, but most of them operate in the same way.

As shown in Figure 3, a special audio oscillator, operating at 2400-Hz in conjunction with a "divide-by-2" IC chip, is keyed by the computer to produce the correct string of 1s and 0s that forms the data. This string of serial tones is then plugged into the cassette recorder's microphone or auxiliary input jack and recorded on the cassette tape. During playback the audio tones are picked off the cassette tape, amplified by the tape recorder system, and fed via the earphone connector to the tape recorder input connector on the cassette interface board. The recording and playback speed of this tone system can be almost anything that the user chooses, which is one of the troubles with the whole idea. There are so many different audio-cassette recording systems on the market that a hobbyist is hard put to make up his mind about the one to use. When planning to buy or build, the speed of recording and playback should not be the major consideration. The user should find out which system has the greatest software support among the people with whom he is likely to exchange programs. In fact, he may have a 300-baud tape interface with which to interchange tapes and another that operates at a much faster speed for use with his own computer and tape player. The slow-speed 300-baud system permits tape interchange over telephone lines by using modems (telephone couplers) and also allows for variations in the speed of low-cost players. If you are going to use your tape recorder only to record data and play them back to your own computer, you can select any speed that works for you.

Modems

Although not used by many hobbyists, the modem (acronym for *mo*dulator-*dem*odulator) permits one computer to "talk" to another or one terminal to pass data to a computer or to another terminal via conventional telephone lines or other limited bandwidth communication channels. If you intend to do any time sharing (i.e., in which several terminals are connected to one computer via phone lines), the modem is the only way to go. All you need to do in most cases is dial the telephone number of the computer and when it "answers," insert your telephone handset into the acoustic coupler of the modem, as shown in Figure 4. Once this is done you can start using your terminal as if it were directly connected to the computer. In this way a single computer can be made to service a number of "subscribers," and although a large number of terminals can be using the computer at the same time the computer may be switching around between terminals so fast that each operator will think he is the only one on the line.

Modems use a tone system similar to that employed by cassette tape recorders. Modems can operate in one direction only—called half-duplex—or in both directions at the same time—called full-duplex. Although most modems can operate at 9600 baud, full-duplex operation over an ordinary telephone (bandwidth-limited) usually limits the speed to about 300 baud. Many modems employ two different sets of audio tones called high/low bands: one for transmitting and the other for receiving.

Physically, a modem looks like a small rectangular box with two large holes in its upper surface spaced for the telephone handset microphone and earpiece. Each hole is surrounded by a soft rubber seal to form a tight fitting around earpiece and microphone to block out extraneous noises. Some modems have front-panel switches that allow half-or full-duplex operation and tape recording of data.

Actually there are two ways to get a telephone line into a modem. One is a direct-wire connection which uses a coupler supplied by the telephone company. Most hobbyists, however, go the acoustic coupling route shown in Figure 4, which means that no direct connections are made to the phone line. A small speaker talks into the phone microphone while a small microphone listens at the earphone end.

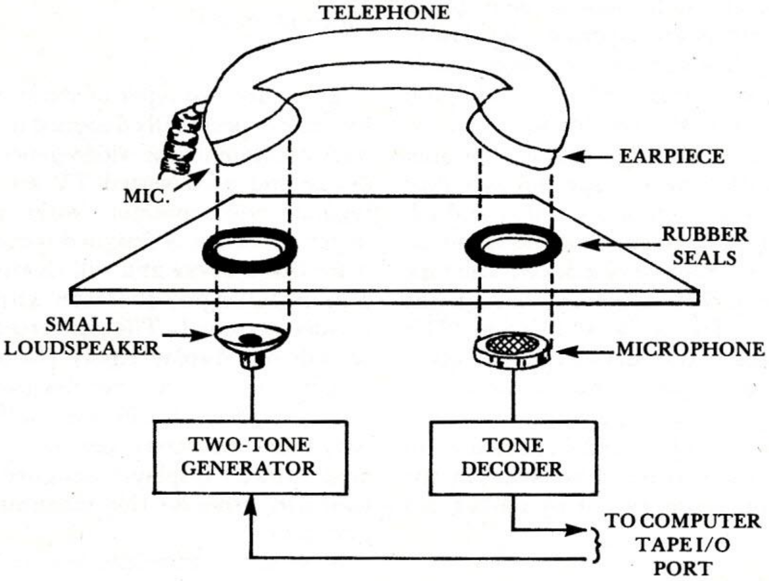


Figure 4
A modem or acoustic coupler. In this case an ordinary telephone line links two computers.

Floppy Disk

This relatively expensive approach to mass storage combines tape recording and audio record techniques. The disk is a large piece of magnetic recording medium cut into the shape of a record. It revolves on a center spindle just as a record does, but the recording/pickup head does not move in a spiral fashion. Instead the magnetic head moves across the disk in a straight line and the tracks on the disk are not connected. The disk controller electronics cause the head to move to any particular track to record or pick up data. Typical disks can store many hundreds of thousands of digital words and some can store millions.

There are two types of disk—hard and floppy. A hard disk is stiff and somewhat like a conventional plastic phonograph record. The floppy disk is made of thinner material and is "floppy." Both types of disk usually come in protective packaging and the raw disk is never exposed to human contact (finger oils will injure the disk's sensitive surface). Recording and playback are usually made through a small window in the protective shield.

Other I/O Devices

Until recently the only way to "read" paper tapes into a computer was to use a relatively expensive ASR-33 Teletype equipped with a tape reader (also a tape perforator). To read a paper tape the tape is inserted into a special tape guide and a plastic tape retainer is clamped over it. When the TTY switch is placed in the tape read position, a sprocket wheel pulls the tape over a set of seven slender metal fingers that come up under it. If there is a hole in the tape immediately above any of the fingers, the finger passes through the tape. In doing so the bottom of the metal finger makes contact with an electric bar in the tape reader to complete an electric circuit similar to that of a keyboard key being depressed. The ASCII combination of

ROMtutorial ROMtutorial

Scrolling: A method of displaying data on a TV screen in which the display "rolls up" one line when a line has been filled. A scrolling system allows you to recall the data that was on the screen as if the screen were a window opening on a long roll of text. By pressing keys on the terminal, you can move the text around behind the window as if it were a scroll of paper. This makes it easier to read than the "paging" display in which the entire screenful disappears after the last line is written.

PROM: Programmable Read Only Memory. A computer memory that lets a user store a program or data inside by burning out sections of the memory (chip).

Analog-to-digital converter: An electronic device that looks at a voltage which can have a continuous range of values and converts it to a code consisting of digital on-or-off signals.

X-y joystick: A lever which normally stands straight up, but can be moved out of vertical in any direction. The *x* direction is along the right-to-left line; the *y* direction is along the forward-to-back line. A computer hooked to a joystick could move a spot anywhere on a video display screen in response to the stick deflection. A common use of a joystick is to move a pointer or window called a cursor around on a screen.

the contacting fingers then forms the word to be transmitted or typed on the TTY.

This means that the relatively weak paper tape has to withstand the repeated impacts of the metal fingers probing for their holes; unfortunately this action often tears it at its weak points. Thus erratic data can be introduced into the program played back by the tape. The action of the sprocket wheel that pulls the paper tape along can also damage it.

Another problem introduced by the tape reader, already discussed, is speed. For the most of us the 110-baud of the TTY is too slow. It can take as long as fifteen minutes to enter a typical 8K BASIC. If you have a longer program, you may wait forever.

Some commercial tape readers can operate at very fast speeds, but they are expensive. Unless you happen to find one at a low price at a flea market, you won't see many of them around.

The commercial tape readers that operate at high speeds use the optical reader system in which a light is concentrated on a narrow section of the moving tape. A set of light-sensitive cells is arranged in a row across the width of the tape and as the tape moves the light shines through the holes into the light cells, but not through the unbroken paper. There are eight sensors for data and one for the sprocket hole. Most of these optical tape readers are expensive, but Oliver Audio Engineering, a West Coast firm, has introduced a low-cost unit for the hobbyist and home.

To use this tape reader the user threads the paper tape through tape guides at the top of the reader and adjusts a light source to shine down on the tape-sensor combination. The tape is then pulled manually through the guides as fast as the user desires. This is a parallel-port device that will operate as fast as your computer will accept the data. By using this device 8K BASIC can be loaded in a matter of seconds and there is no wear on the tape unless you tear it by pulling too fast.

Video Displays

General

Video displays are the TV-like devices that you see working with computers and displaying data on a CRT screen. There are several types: first, the video terminal, which is a self-contained unit equipped with a keyboard, video character generator, and CRT display. The video terminal is connected directly to the computer or by a modem via telephone lines to a remote machine. They are the video equivalent of a TTY or other printer terminal. Some low-cost video terminals are suitable for use with a personal computer. Southwest Technical Products makes the CT-64 Video Terminal Kit for $325, Micro-Term, Inc. makes the Act-1, a complete terminal less the CRT for $400, and Micon Industries makes the "Captain Video" series. Several of the manufacturers of commercial video terminals now offer kit versions of their regular units. The Lear Siegler ADM-3A kit and the SOROC Technology IQ 120 kit are examples. When assembled, they are equal to the commercial versions. Their cost is less than $1000 in kit form, a price that is low by industry standards but still high for a hobbyist.

The answer to this problem has been the development of video display modules that convert the digital output of the data bus into alphanumeric video signals. These modules are boards that plug directly into the computer bus and output video signals to an external video monitor or converted TV set.

Video Monitors

There are two types of video monitor: those specifically designed to work with a computer or video-generating device and a converted TV set. The regular video monitor works much better because it is designed specifically for this purpose and will clearly display the full eighty-character output of a video terminal. The converted TV set will not display eighty characters clearly because it was not designed for this application. It will work well with sixty-four characters per line; thus most video displays designed for hobbyists generate this maximum of sixty-four.

Some hobbyists neither wish to buy a video monitor nor convert a black and white TV and therefore attempt to use an RF converter and an unmodified TV set. Several versions of one or two transistor TV transmitters or converters are available. They accept the composite video signals from the video display module in the computer and convert them into an RF signal that will pass from the set's antenna terminals through the TV front end into the video circuits. Several things are wrong with this idea. First, most of these kits are not type-approved by the FCC and are therefore illegal. Second, aside from graphics displays, they don't work very well. If you want to convert a TV set into a monitor, use a black and white set with an isolation power transformer and perform the conversion according to instructions in one of the magazines or in Don Lancaster's excellent *TVT Cookbook,* published by Howard W. Sams.

Several TV manufacturers will soon be offering models with video input connections and switches to disconnect the antennas. They are doing this for the TV game market and their own video tape recorders, but the computer hobbyist will benefit.

Video Display Modules

The first video display module suitable for use by the hobbyist was the VDM-1 produced by the Processor Technology Corporation. This unit plugs into the S-100 bus and generates sixteen lines of sixty-four characters across the screen. It can produce white characters on a black background or black characters on a white background. In addition, it scrolls like an expensive video terminal and is very fast, for it can fill the screen with data in an instant. It has its own memory and the user can control the writing speed on the screen. The VDM-1 also has limited graphics capability and is considered as the standard with which other video displays are compared. Similar boards are made by other manufacturers and advertised in hobby publications.

PolyMorphic Systems makes a video display for its Poly-88 that is also used in other S-100 bus systems. This unit has a parallel input port suitable for the connection of a keyboard. The Poly video module can therefore be the only I/O board in a video output computer.

A new entry on the video display market is the Merlin, two boards that plug into the Altair/S-100 bus and generate both alphanumeric characters and complex graphic patterns. The Merlin also has PROMs that contain data monitor software and can be programmed to produce any kind of character desired.

Other computer manufacturers have provided their machines with video capabilities. The entire Digital Group concept features video output and their video displays, often used in hobbyist "home brew" systems, are excellent.

OSI, Apple Computer, MicroMind, and Wave Mate, also use video-module-based systems of high quality. Because the cost of *hard copy* terminals exceeds the cost of most personal computers, it seems that video display provides the most cost effective way to present data.

The TV Dazzler

Cromemco Inc. has produced one of the most unusual video displays ever invented. It is called the TV Dazzler and it plugs into the Altair/S-100 bus. Using programs provided by Cromemco or developed by the operator, the Dazzler provides fantastic displays of full color graphics on a color TV receiver or monitor. It can be programmed to produce alphanumerics or color patterns and introduces the element of creative art into the home computer field. Cromemco also produces an analog-to-digital and digital-to-analog converter board called the D-7A, which can be connected to "joystick" controls, also supplied by Cromemco, and installed in the computer to control the color patterns on the Dazzler in real-time applications. The hobbyist can use one of the approved RF converters with his color TV to eliminate the difficult job of converting a color set.

TV Display

Most computers can be provided with a built-in or plug-in video display module to convert computer data to alphanumerics for display on a TV-like screen.

There are four ways in which this visual display can be accomplished:

The first is to have the computer's video system built within the computer enclosure. This is done in most commercial and a couple of the newer hobby systems. Obviously the cost of the video display is added to the cost of the computer system.

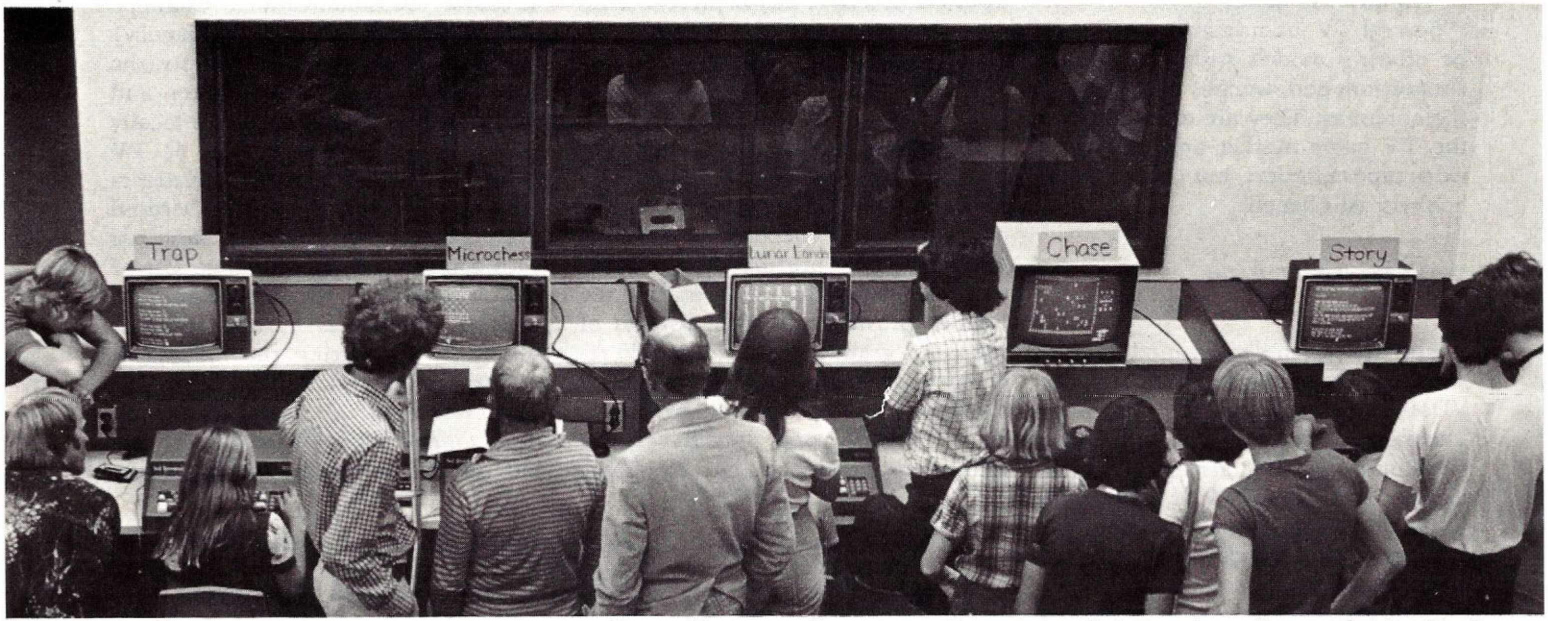
In the second approach, which is the most common and the easiest to implement, the computer video display is terminated with a video coaxial cable that can be plugged directly into a commercial video monitor. Video monitors come in all screen sizes and can be purchased at any computer hobby store.

The third approach uses an FCC (Federal Communications Commission) approved RF (radio frequency) modulator whose input accepts the video cable from the computer and whose output can be tuned to a locally unused low-band (2 through 6) TV channel. These small RF modulators must be approved by the FCC because they transmit RF that can interfere with commercial broadcasting. Use of such a modulator means that any TV receiver can act as a video monitor simply by connecting the modulator output directly to the TV receiver antenna terminals and tuning them both to an unused channel. Because of certain electronic problems (primarily bandwidth problems), do not expect a sharp and crisp image. For most hobby uses, however, this type of display works adequately and is the lowest priced approach to video display.

TV receiver conversion is not recommended; it can blow your whole system.

The last approach is not recommended for the beginner. It involves rebuilding a conventional TV receiver so that it uses only its internal video amplifiers and sweep sections. The selected TV receiver must have power-line isolation in the interests of safety and special video connectors have to be added. This conversion must be done by a qualified TV technician, for there is considerable danger of shock and possible electrocution. There's also a good chance it will blow your whole system—which would truly make TV receiver conversion the last approach. As with anything else, when it comes to computer peripherals, you can only economize so much, before you lose more than you gain. ▼

COMPUTER COUNTRY An Electronic



Where computers are not fragile or threatening machines is where kids belong

David Fox, recently out of a job as a counselor, decided he would rather try a new line of work: running a computerized Disneyland. His wife, Annie, agreed. So they set out to put their idea into action.

Just one year after the dream was hatched, the Marin Computer Center opened on a hilltop near San Francisco, with nine Sol-20 microcomputers, some videotape equipment, and a lot of fascinated kids of various ages. It's not really a Disney-scale futureworld, but it is the first of the "second generation" computer game centers. (The Community Computer Center in Menlo Park, California, was the only one of the first generation.) The Marin Computer Center's purpose is "to introduce advanced technology for learning and entertainment," and none of the people running it has had to become an expert in computers. They've received help from amateur computer enthusiasts in the area and have also benefited from the latest stages of microcomputer development.

David and Annie researched two similar computer game centers in the local area (CCC and the Lawrence Hall of Science at the University of California) to find out how things were done there. In both cases they found one computer at each place, with complicated hardware and programs to allow several different users to play games through terminals. They always found a few people at each place who knew how to repair the equipment in case of breakdowns. Highly skilled, poorly paid, these people were usually worried that a

Photographed by Lee Felsenstein

Jungle Gym for Kids

by Lee Felsenstein

Community computer centers may well be the bingo parlors of the future.

really big breakdown would exceed their capabilities.

They also saw shaking, clattering Teletypes which always needed fixing, and the computer itself installed behind walls for protection from curious fingers. Not quite what the Foxes had in mind for their game center. Annie in particular wanted a pleasant environment for kids, in which the computers were not fragile or threatening "machines."

The Foxes researched the new computers and microcomputers just beginning to be marketed. They found prices were down quite a bit from what earlier centers had had to pay, and that a whole computer could now be bought and set up for about the same price as terminals alone used to cost. The new microcomputers didn't need terminals—they had their own keyboards and could make a slightly altered television set (or "video monitor") display the words and numbers (as well as drawings) they produced. And since each one was a separate computer, a failure of one did not mean all the others stopped working.

But what about the money to buy the computers? Not to mention paying salaries to the Center's employees until the cash flow from the operations could handle it. The Foxes wanted to ask for a large foundation grant, but they soon found out that it takes an awful lot of time and work to get that kind of "free money." Instead they joined forces with Sue Long, a student at the Harvard University Business School, and decided to tackle a bank. Sue knew enough about managing a business to convince the bank a loan was in order.

They also talked to the local San Rafael School District and got them to agree to send classes of kids to the Center to learn how to use computers. The school district could get money from the federal government for such a program, to pay for the services provided by the Center. The school district also agreed to let the Center (run by a non-profit corporation called Ulenar) rent a library and two classrooms in a school which had run out of kids and was closed.

When it came time to decide definitely which computer to get, the Foxes asked a lot of people for opinions. Finally they decided on the Sol-20s, which look like typewriters and have wooden sides. Annie said that they were her choice because they didn't look like computers, ran silently, and had "warm colors." They made a deal with a local computer store so that the store would get all the computers, put them together with the TV monitors, and would service them on a continuing basis. They were able to save money by doing things this way instead of buying the computers direct and they also had some expert assistance available.

David came to a Homebrew Computer Club meeting to get in touch with people there who had Sol computers and knew how to make interesting programs run on them. These new acquaintances provided both a lot of help and lots of game programs. Most of them hadn't known any more about computers than David did when they got theirs, so they had learned the hard way about things to do and not to do to make their computers work.

The Foxes also found some videotape equipment and a large-screen TV display which would work with either the video cameras and tape recorders or with the computers. They stocked a shelf with books to read, books to borrow, and magazines (including *ROM*), and books for sale. Then they cleaned up everything and announced the opening date.

GAMES COMPUTERS PLAY

When the Marin Computer Center opened, these games were up and running:

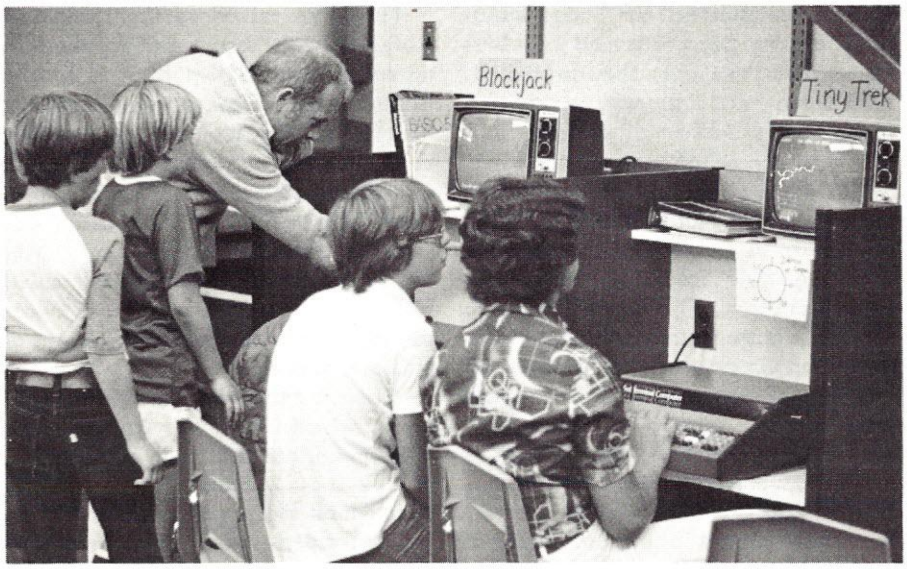
Trap	Guess the computer's secret number. From the PCC games book.
Microchess	A chess program from Microware Ltd., 27 Firstbrooke Rd., Toronto, Ontario, Canada M4E 2L2.
Lunar Lander	Control a landing on the moon. From the PCC games book.
Chase	Chase the computer through a maze. From *dr. dobb's journal.*
Story	Give the computer some favorite words and get them back in a story the computer tells you. Written in the PILOT language by Dr. John Starkweather and published in *dr. dobb's journal.*
Trek-80	Pilot the starship Enterprise complete with viewscreen. From Processor Technology.
Tiny Trek	A smaller version of *Star Trek.* Written for Palo Alto Tiny BASIC.
Blackjack	The card game, if you trust the computer to deal. Also written for Tiny BASIC.

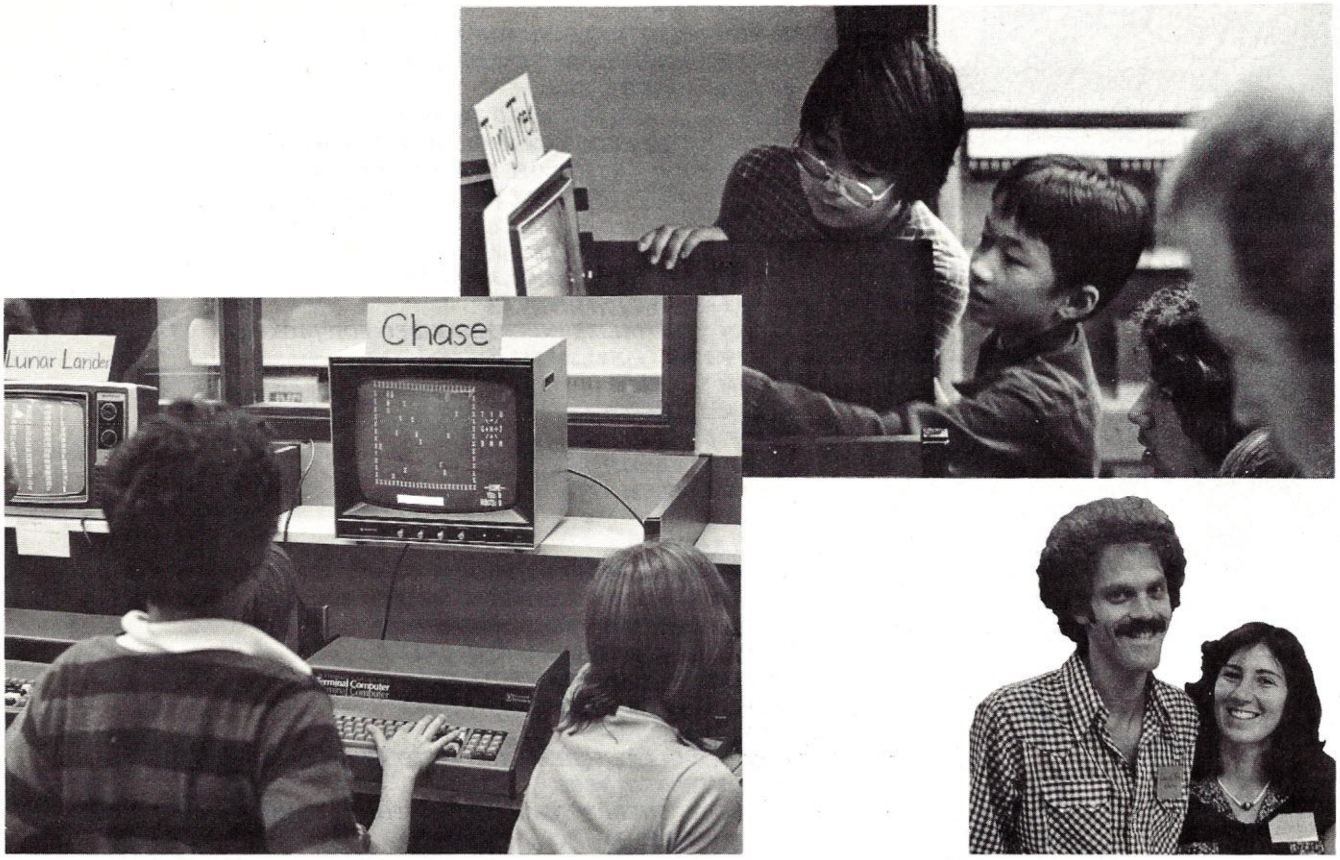
On the first day some two hundred people came, mostly families with kids. There was no charge for using the computers at the opening; later this would change, to a fee allowing anyone to use a computer for $2.25 per hour (no matter how many friends came along). If one had paid for a membership in the Center, the rate went down to $1.50. Groups and parties could use the computers for $1.50 per child for one and a half hours.

After the opening, the Center began teaching kids from the school district in an eight-week course starting with games and continuing to let the kids learn to write their own programs and try them out. The future calls for adult courses as well, and the Center will let people use the computers to work on any program they want to develop. On the very first day, a college student came by to ask whether he could use the computers to do some calculations for a study project. For just this sort of need the Center will be open once a week so people can drop in without having to make reservations.

In the future David and Annie want to see more realistic games developed for the computers—games which several people could play together, each with his own computer. They also want to involve handicapped people with computers as a means of letting them deal with the rest of the world on a more equal basis.

Community computer centers may well be the bingo parlors of the future. If you or your school has been considering such a project, it would probably pay to check in with the Foxes first. They can be reached at The Marin Computer Center, 70 Skyview Terrace, Room 301, San Rafael, CA 94903. Phone: (415) 472-2650. ▼





FIRST GENERATION

Established in 1972 as People's Computer Center, the Community Computer Center was the first computer education and game-playing parlor not part of a university or science museum. It occupies a former grocery store on a quiet street in Menlo Park, California, about twenty-five miles south of San Francisco, quite close to Stanford University.

At the time CCC was growing up, minicomputers were the smallest, most available computers on the market. The PDP-11, which they eventually obtained, would cost about $25,000 if it were bought today. Sixteen users can work at terminals wired up to the computer, running different programs in BASIC language for each terminal. This kind of operation is called "time sharing."

CCC is owned by a non-profit corporation and earns its keep by charging a small fee for use of the computer. They arrange with schools to bring in classes of kids, in which case the school pays the rate. In afternoons and some evenings, anyone can drop in and pay to use the computers, either to play games or to write and test their own programs. There are a lot of regular, non-computer games there as well, in case the terminals are full or the computer is "down" (not working).

One of the disadvantages of any single-computer time-sharing system is that if any part of the computer stops working, the whole operation stops until it is fixed. This means that someone has to be available all the time who knows how to fix the computer and all of its peripheral devices. The computer itself has to be protected in its own air-conditioned room, even though the people who run CCC would like to let kids touch the computer so it wouldn't seem so strange.

Several computer games have been developed at CCC; these are published in a widely-distributed book called *What To Do After You Hit Return*. CCC also makes punched-paper-tape copies of games for people who have a Teletype and access to some other computer.

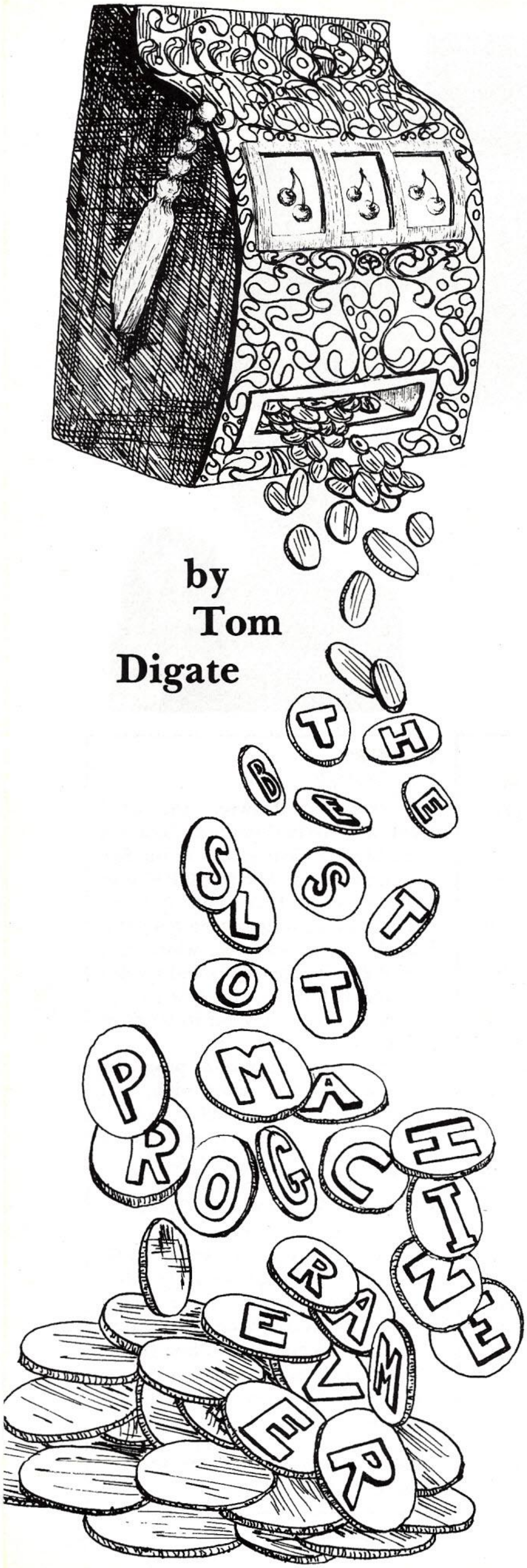
CCC's address is 1921 Menalto Ave., Menlo Park, CA 94025. Phone: (415) 326-4444.

LOOP'S LO*OP

Somewhere between the CCC and the Marin Computer Center is the LO*OP Center, also in the San Francisco area. The LO*OP Center was set up by Liza Loop, a determined young woman who bought a DEC classic computer, which is a PDP-8 with floppy disks and a video terminal. It's not exactly a microcomputer, certainly not in terms of its price.

Liza runs educational programs with elementary, junior high, and high school kids, as well as providing a meeting place for the Sonoma County Microcomputer Club, of which she has been president. The LO*OP (for Learning Options—Open Portal) Center has not become as widely known as CCC, but has been in operation for a few years, and qualifies as the second public access computer games-and-education center in the "first generation" of centers—those with one computer and time-sharing access to others.

The LO*OP Center can be reached by writing to LO*OP Center, Cotati, CA 94928.



**by
Tom
Digate**

```
10 REM * SLOT MACHINE FOR PTC BASIC5 *
20 REM * WRITTEN BY TOM DIGATE *
30 REM * AUGUST, 1977
35 SET S=0: REM SET SPEED TO FAST
36 REM * THIS PROGRAM SHOULD BE RUN WITH
37 REM * THE CURSOR OFF FOR BEST DISPLAY.
40 REM * INITIALIZE SLOT REELS AND PAYOFFS
50 DATA 2,4,2,5,1,3,4,5,3,4,1,2,3,1,6,2,5,1,3,2
60 DATA 3,1,2,3,1,2,3,1,4,4,1,2,3,6,3,1,2,3,5,1
70 DATA 2,3,4,5,2,5,4,3,6,4,2,3,5,2,2,3,4,2,3,2
75 DATA 3,5,6,8,10,15,18,20,200
76 REM * INITIALIZE INVERTED VIDEO FOR JACKPOT
77 DATA 202,193,195,203,208,207,212
80 REM * CHERRY 1
90 REM * ORANGE 2
100 REM * BELL   3
110 REM * LEMON 4
120 REM * PLUM   5
130 REM * BAR      6
135 CLEAR
140 DIM R1(20),R2(20),R3(20),P(9)
150 FOR I=1 TO 20: READ R1(I): NEXT
160 FOR I=1 TO 20: READ R2(I) : NEXT
170 FOR I=1 TO 20: READ R3(I): NEXT
180 FOR I=1 TO 9: READ P(I): NEXT
182 FOR I=1 TO 7: READ J1(I): NEXT
185 RESTORE
190 REM * CLEAR SCREEN AND PRINT WINNING COMBO'S
200 PRINT "&K&[&B&6"
210 PRINT "CHERRY";TAB(12);"ANYTHING";TAB(24);"ANYTHING";
220 PRINT TAB(36);P(1);" COINS"
230 PRINT "CHERRY";TAB(12);"CHERRY";TAB(24);"ANYTHING";
240 PRINT TAB(36);P(2);" COINS"
250 PRINT "ORANGE";TAB(12);"ORANGE";TAB(24);"BAR";
260 PRINT TAB(36);P(3);" COINS"
270 PRINT "BELL";TAB(12);"BELL";TAB(24);"ORANGE";
280 PRINT TAB(36);P(4);" COINS"
290 PRINT "LEMON";TAB(12);"LEMON";TAB(24);"LEMON";
300 PRINT TAB(36);P(5);" COINS"
310 PRINT "PLUM";TAB(12);"PLUM";TAB(24);"BAR";
320 PRINT TAB(36);P(6);" COINS"
330 PRINT "ORANGE";TAB(12);"ORANGE";TAB(24);"ORANGE";
340 PRINT TAB(36);P(7);" COINS"
350 PRINT "PLUM";TAB(12);"PLUM";TAB(24);"PLUM";
360 PRINT TAB(36);P(8);" COINS"
370 PRINT "BAR";TAB(12);"BAR";TAB(24);"BAR";
380 PRINT TAB(36);"JACKPOT —";P(9);" COINS";
381 REM CLEAR FIRST 5 LINES
382 PRINT "&N": PRINT : PRINT : PRINT : PRINT : PRINT
390 REM REPOSITION CURSOR AT LINE 1
400 PRINT "&N";
401 PRINT "* * * * * * COMPUTER SLOT MACHINE * * * * * *": PRINT : PRINT
410 INPUT "TYPE '0' AND 'RETURN' TO PLAY----  "A
412 PRINT "&N": PRINT : PRINT
418 GOTO 2000
419 *
420 REM * VALUE CALCULATION SUBROUTINES *
430 N=1
440 V1=INT(20*RND(0))+1
450 R=R1(V1)
460 GOSUB 5000
470 RETURN
600 N=12
610 V2=INT(20*RND(0))+1
620 R=R2(V2)
630 GOSUB 5000
640 RETURN
800 N=24
810 V3=INT(20*RND(0))+1
```

Illustrated by Cindy Hain

```
820 R=R3(V3)
830 GOSUB 5000
840 RETURN
1990 REM * CHECK TO SEE IF ABORT REQUESTED
2000 IF A<>0 THEN PRINT “&K”
2010 IF A=0 THEN 2015
2011 PRINT : PRINT “TYPE ‘RUN’ AND HIT ‘RETURN’ TO PLAY”
2012 GOTO 65387
2013 REM * REPOSITION CURSOR
2015 GOSUB 4500
2016 REM * DISPLAY REELS 1,2, +3
2020 FOR I=1 TO 30
2030        GOSUB 430
2040        GOSUB 600
2050        GOSUB 800
2055        GOSUB 4500
2060 NEXT
2065 REM * DISPLAY REELS 2+3
2070 FOR I=1 TO 20
2071        R=R1(V1)
2072        N=1
2073        GOSUB 5000
2080        GOSUB 600
2090        GOSUB 800
2095        GOSUB 4500
3000 NEXT
3005 REM * DISPLAY REEL 3
3010 FOR I=1 TO 10
3011        R=R1(V1)
3012        N=1
3013        GOSUB 5000
3014        R=R2(V2)
3015        N=12
3016        GOSUB 5000
3020        GOSUB 800
3025        GOSUB 4500
3030 NEXT
3040 R=R1(V1)
3050 N=1
3060 GOSUB 5000
3070 R=R2(V2)
3080 N=12
3090 GOSUB 5000
4000 R=R3(V3)
4010 N=24
4020 GOSUB 5000
4030 PRINT
4040 GOTO 7000
4400 GOTO 2000
4490 REM * REPOSITIONING SUBROUTINE IS BELOW
4500 PRINT “&[&B&B&[&A&A”: RETURN
5000 REM * BEGIN PRINTING SUBROUTINES HERE *
5050 IF R=1 THEN PRINT TAB(N);“CHERRY”;
5060 IF R=2 THEN PRINT TAB(N);“ORANGE”;
5070 IF R=3 THEN PRINT TAB(N);“BELL”;
5080 IF R=4 THEN PRINT TAB(N);“LEMON”;
5090 IF R=5 THEN PRINT TAB(N);“PLUM”;
6000 IF R=6 THEN PRINT TAB(N);“BAR”;
6010 RETURN
7000 REM * CALCULATE WINNINGS OR LOSINGS
7005 REM * SET A,B,C EQUAL TO 6 IF YOU WANT
7006 REM * TO SEE JACKPOT PRINTED IN INVERSE VIDEO
7010 A=R1(V1)
7020 B=R2(V2)
7030 C=R3(V3)
7080 IF A=B THEN 10000
7090 IF A=1 THEN 8050
8000 W=W—1
8010 PRINT TAB(14);“You lost 1 coin----”
```

```
8020 GOTO 50000
8049 REM *******
8050 W=W+P(1)
8060 PRINT TAB(14);“You won”;P(1);“ coins----”
8070 GOTO 50000
10000 REM * DETERMINE WINNING COMBINATION
10010 IF B<>C THEN 20000
10015 REM * NOW A=B=C
10020 IF A=1 THEN 20000
10021 IF A=4 THEN W=W+P(5) : A=5: GOTO 10060
10022 IF A=2 THEN W=W+P(7) : A=7: GOTO 10060
10023 IF A=5 THEN W=W+P(8) : A=8: GOTO 10060
10024 IF A<>6 THEN 8000
10040 REM * CHECK FOR JACKPOT
10050 W=W+P(9) : A=9: GOSUB 60000
10060 PRINT TAB(14);“You won”;P(A);“ coins----”
10070 GOTO 50000
20000 REM * NOW A=B AND B<>C
20010 REM * CHECK FOR CHERRY
20020 IF A<>1 THEN 21000
20030 W=W+P(2) : A=2
20040 GOTO 10060
21000 REM * CHECK FOR ORANGE-ORANGE-BAR
21010 IF A<>2 THEN 22000
21020 IF C<>6 THEN 8000
21030 W=W+P(3) : A=3: GOTO 10060
22000 REM * CHECK FOR BELL-BELL-ORANGE
22010 IF A<>3 THEN 23000
22020 IF C<>2 THEN 8000
22030 W=W+P(4) : A=4: GOTO 10060
23000 REM * CHECK FOR PLUM-PLUM-BAR
23010 IF A<>5 THEN 8000
23020 IF C<>6 THEN 8000
23030 W=W+P(6) : A=6: GOTO 10060
50000 REM * PRINT FINAL RESULTS *
50010 PRINT “* * * * * * * You now have ”;W;“ coins * * * * * * *”
50015 REM * DELAY BEFORE ACCEPTING NEXT PLAY
50020 FOR I=1 TO 3000
50030 NEXT
50040 GOTO 381
50045 REM *******
60000 REM * JACKPOT *
60010 REM * DO SPECIAL DISPLAY *
60020 FOR I=1 TO 5
60030        PRINT “&[&B&D&[&A&A* * * * * * * JACKPOT * * * * * * *”;
60040        FOR J=1 TO 500: NEXT J
60050        PRINT “&[&B&D&[&A&A* * * * * * * ”;
60051        FOR J=1 TO 7: C1=J1(J) : GOSUB 62000: NEXT J
60055        PRINT “ * * * * * * *”;
60060        FOR J=1 TO 500: NEXT J
60070 NEXT I
60080 GOTO 65380
61999 REM *** OUTPUT INVERTED VIDEO SUBROUTINE ***
62000 C1=C1: REM * DUMMY STATEMENT
62005 REM * SETUP LINK TO SOUT IN SOLOS
62010 S=49177
62011 REM * SETUP ESCAPE CODE IN B REGISTER
62020 B=ARG(27*256)
62030 M=CALL(S)
62035 REM * SETUP CODE 5 IN B REGISTER
62040 B=ARG(5*256)
62050 M=CALL(S)
62055 REM * SETUP ACTUAL CHARACTER IN B REGISTER
62060 B=ARG(C1*256)
62070 M=CALL(S)
62080 RETURN
65380 PRINT “&[&B&D&[&A&A* * * * * * * JACKPOT * * * * * * *”
65384 RETURN
65387 SET S=0: REM * EXIT WITH SPEED AT 0
```

THE Better Health

by Karen E. Brothers

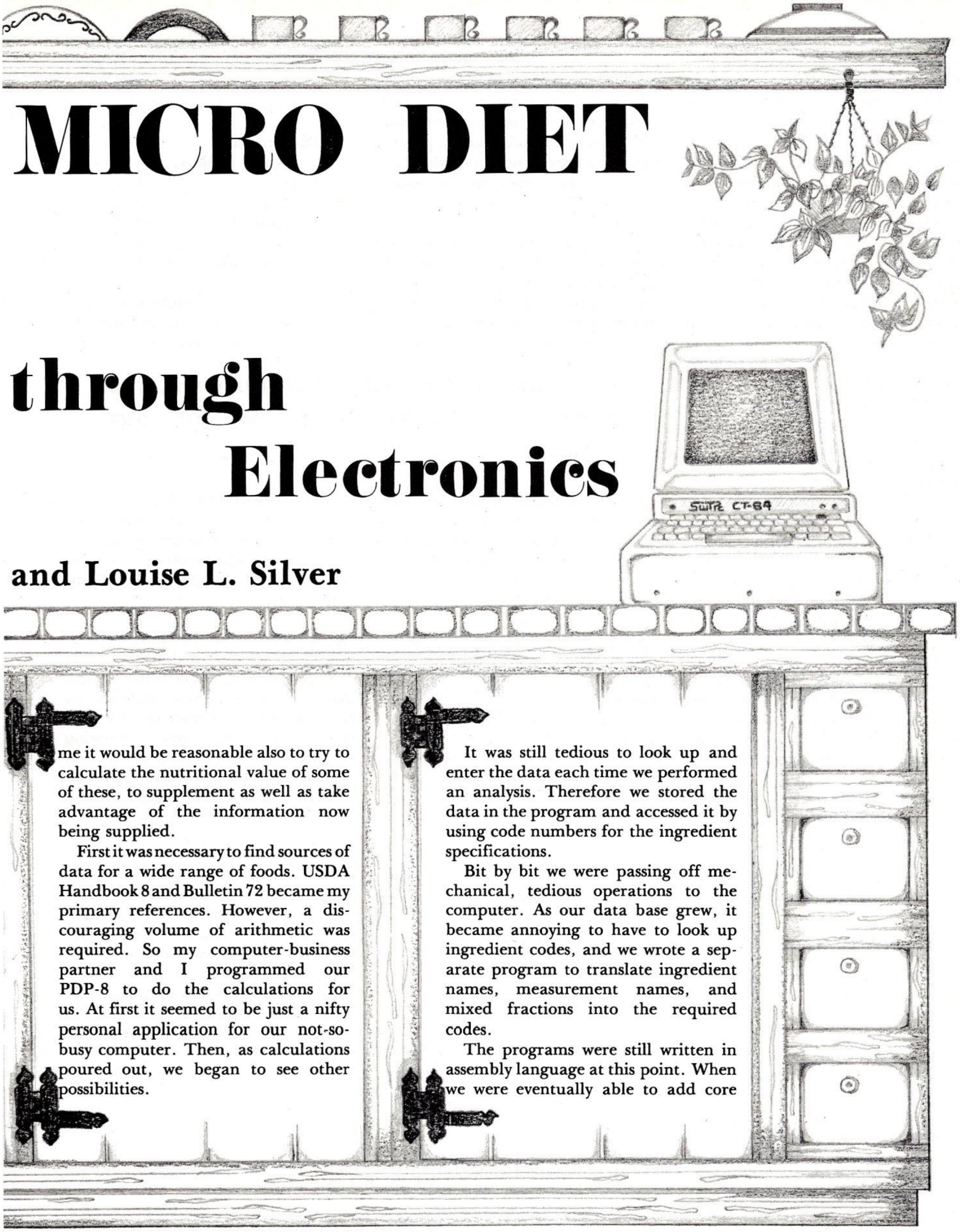
Several years ago nutrition began getting media coverage. That's about when I got interested. The FDA started requiring that nutrition and ingredient information be provided to consumers, and the advent of unit pricing provided a means of measuring the bang for the buck.

This was great; a customer could pick up a box or can from the supermarket shelf, read the nutritional information printed on the label, and determine the food value of the dish when served as directed. But, I thought as I scurried up and down the store aisles, what about my own recipes, or, for that matter, any foods not so informatively packaged? It seemed to

Illustrated by Linda Smythe

MICRO DIET through Electronics

and Louise L. Silver



me it would be reasonable also to try to calculate the nutritional value of some of these, to supplement as well as take advantage of the information now being supplied.

First it was necessary to find sources of data for a wide range of foods. USDA Handbook 8 and Bulletin 72 became my primary references. However, a discouraging volume of arithmetic was required. So my computer-business partner and I programmed our PDP-8 to do the calculations for us. At first it seemed to be just a nifty personal application for our not-so-busy computer. Then, as calculations poured out, we began to see other possibilities.

It was still tedious to look up and enter the data each time we performed an analysis. Therefore we stored the data in the program and accessed it by using code numbers for the ingredient specifications.

Bit by bit we were passing off mechanical, tedious operations to the computer. As our data base grew, it became annoying to have to look up ingredient codes, and we wrote a separate program to translate ingredient names, measurement names, and mixed fractions into the required codes.

The programs were still written in assembly language at this point. When we were eventually able to add core and secondary storage to our machine, we rewrote and consolidated the programs into a single BASIC program. While we were at it, we moved the nutritional data out of the program and into a file, improved the output format, and made it easier to add or subtract nutrients from the analysis.

We then purchased a magnetic tape compiled by the government, with nutrient data for 2,400 foods. However, the data was incomplete for our purposes. It was based on weight, and lacked the conversion factors we needed to specify ingredients listed in recipes as whole or by volume. So we researched, computed, and added these factors to the data base; now hundreds of different ingredients can be specified by the piece, by volume, or by weight.

The research necessary to implement our recipe-analysis program proved to be an educational experience, as did the exposure to nutritional data gained by performing the many analyses. The more we learned about nutrition, the more applications we found for our software. We have used it to pinpoint nutritional inadequacies in our own diets, to help us improve the food value of our recipes, and to determine which recipes simultaneously satisfy special dietary requirements such as low carbohydrate, low sodium, and low cholesterol. By ascertaining the grams of protein per pound and utilizing unit-pricing figures, we could compare the relative economy of protein sources: peanut butter is one of the cheapest, and tuna beats hamburger in cost per gram of protein.

In the future we can see other possibilities. If the data base could be maintained efficiently, cost-per-serving could be added to the analysis. The major difficulty continues to be keeping up with changing food prices.

Another possibility we look forward to is the automatic addition of recipe analyses to the data base. In time this would make possible some fairly sophisticated menu planning. Or a system could be developed incorporating recipe analysis and kitchen inventory, so menu planning could be taken over wholly or in part by the computer. This might involve developing parameters for food compatibility, efficient use of resources (such as oven heat), family preferences and dietary restrictions, even what foods (such as peaches in summer) are in season. Dishes high in certain nutrients could be sought out to complement dishes low in those nutrients.

A micro can do some fairly sophisticated menu planning, recipe analysis, kitchen inventory, and even energy saving.

The Personal Nutritional Analysis Program produces, based on the list of ingredients in a recipe or menu, an analysis reporting the amounts of calories, protein, carbohydrates, fats, sodium, calcium, iron, vitamin A, vitamin C, thiamine, niacin, and riboflavin in the entire recipe and in each serving. In addition, the percentage of U.S. Recommended Daily Allowance per serving is printed for those nutrients for which RDA is relevant.

The more we learned about nutrition, the more applications we found for our software.

To run the program, the following information must first be entered for each recipe (or menu) to be analyzed:

- Number of servings
- Quantity of each ingredient
- Measurement code for each ingredient
- Item code for each ingredient

Used to compute the per-serving nutrient data, the number of servings must be an integer or decimal number. A range (such as "serves four to six") is not permitted; in this case use the average, or some other number within the range. If the RDA percentages are desired for the recipe as a whole, enter "one" as the number of servings. (Entering zero servings causes the program to terminate.)

The quantity of each ingredient must also be a decimal number. For example, if a recipe calls for 1½ cups of milk, type 1.5 as the quantity.

Since the program will not accept the names of measurements (*cups, pounds,* etc.), the following codes must be used:

Measure	Code
Discrete (such as three apples)	0
Cups	1
Tablespoons	2
Teaspoons	3
Pounds	4
Ounces (avdp)	5

All ingredients for which the program has data also have code numbers by which they are referenced, such as 1 for eggs, 2 for milk, and 3 for sugar. The program will accept only these code numbers, not the ingredient by name. It is helpful to be able to refer to a chart of acceptable code numbers and the items they represent.

When run, the program will prompt for the number of servings. You respond with the appropriate number followed by carrriage return. A response of 0 indicates that you're done requesting analyses, and the program terminates.

As it moves on, the program requests ingredient information for one item at a time. In response to the *?* prompt, type the quantity, measurement code, and item code. Separate the numbers with commas and terminate the line with carriage return.

After all the ingredient information has been input, indicate that there are no more ingredients in the recipe by typing zeros: 0,0,0. The program will then respond by printing out the analysis, followed by a request for the number of servings in the next recipe.

Just so that too many cooks don't spoil the broth, the program has error messages to minimize confusion. For instance:

ITEM NUMBER TOO HIGH: n
?

This means the user has entered an ingredient code number which exceeds the number of ingredients provided for in the program. The erroneous item number, *n,* is printed. To correct the error, repeat the input sequence with the correct ingredient code.

Another error message is:

NO MEASUREMENT
CONVERSION: m OF n
?

To commit this error, the user entered measurement code *m* for ingredient number *n;* the program lacks the necessary conversion factor to handle the ingredient as specified. To fix things, retype the input line, using a measurement unit for which the conversion factor for that ingredient has been installed. Weight specifications are always acceptable.

If you see this message:

ILLEGAL MEASUREMENT: n
?

it means you've typed a measurement code *n,* which the program is not equipped to handle. Find the proper code for the measurement unit desired, and reenter the input line.

Although you will want to add to this program, to customize it to your own eating patterns, it's basically as simple as it looks. So eat well and grow healthy. Your computer won't put on weight. ▼

Goodies In, Goodies Out

The analysis output consists of the total amount, amount per serving, and (where applicable) percentage of U.S. RDA per serving of the following nutrients:

Nutrient	Unit	RDA
Calories	calories	—
Protein	grams	45 gm
Carbohydrates	grams	—
Fats	grams	—
Sodium	milligrams	—
Calcium	milligrams	1000 mg
Iron	milligrams	18 mg
Vitamin A	IU	5000 IU
Vitamin C	milligrams	60 mg
Thiamine (B_1)	milligrams	1.2 mg
Niacin	milligrams	15 mg
Riboflavin (B2)	milligrams	1.5 mg

Note that the numerical output should not be considered accurate to more than two significant figures. Also, the program combines nutritional data for the ingredients of a recipe, and ignores such considerations as cooking losses, seasonal variations in foods, or possible reactions between nutrients.

THE CUSTARD CONVERSION

This program is written in a limited version of BASIC, so that it can be implemented on a variety of home computer configurations. All data is maintained within the program; no file operations are necessary. Input is numerically coded to avoid using string functions which might not be available to all users.

As presented, the size of this program is about 2,300 bytes. Removal of the opening remarks (statements 10 through 160) shortens the length to about 1,730 bytes. If desired, the size of the program can be further reduced by shortening the error and prompting messages, removing the nutrient labels from the output format, or reducing the number of nutrients analyzed.

To test the program, enter the same input sequence as the sample run; the results should match to within three significant figures. If not, check that the program has been correctly typed, especially the numbers in the DATA statements.

To facilitate operation of the program, prepare a chart of the ingredients for which the program has data, and their code numbers. The code number for each item is determined by the relative position of its data in the data pool. As distributed, the first DATA statement represents the conversion factors and nutrient data for eggs; therefore the code number for eggs is 1. The ingredient code chart for the program as it stands is simply:

Item	Code #
Egg	1
Milk	2
Sugar	3

To use the program effectively, it will be necessary to add nutritional data for more food items than the three mentioned above. Each ingredient requires fourteen numerical values: two conversion factors and twelve nutrient values. Debugging is easier if all fourteen numbers are included in one DATA statement, but this isn't necessary. If any of the data is unavailable, substitute zero for the missing value. The data pool is an ordered list, and all positions must be filled. However, depending upon the nature of the missing data, substituting zero may affect ease of operation or accuracy of analysis.

SAMPLE RUN

The following sample run produces
the nutritional analysis for a custard made from:

3 eggs
2 cups milk
½ cup sugar

This recipe yields four servings.

```
# SERVINGS?4
INGREDIENTS: QUANTITY,MEASUREMENT,ITEMNUMBER
?3,0,1
?2,1,2
?.5,1,3
?0,0,0

NUTRIENT          TOTAL          PER SVG          % RDA

CALORIES          948.563        237.141
PROT(GM)          36.4117        9.10293          20.2267
CARB(GM)          122.188        30.5469
FATS(GM)          34.3117        8.57793
SODI(MG)          427.745        106.936
CALC(MG)          656.224        164.056          16.4056
IRON(MG)          3.55057        0.887643         4.93174
VITA(IU)          2452.47        613.117          12.2623
VITC(MG)          4.87478        1.2187           2.03156
THIA(MG)          0.311243       0.0778108        6.48398
NIAC(MG)          0.637478       0.159369         1.06252
RIBO(MG)          1.27871        0.319678         21.3129

# SERVINGS?0
```

Keep track of the order in which ingredient data are added to the program, and append the item names to the code chart for the operator. If data are removed, reassign all the item code numbers on the chart, so that each code number accurately reflects the sequential position in the data pool of that food item. The fourteen data fields for each ingredient are:

Conversion factors:

1. Absolute weight in grams, for countable discrete items (such as eggs)
2. Density in grams per milliliter, for volume measurements

Nutrients per 100 grams of food item:

3. Calories
4. Protein (in grams)
5. Carbohydrates (in grams)
6. Fats (in grams)
7. Sodium (in milligrams)
8. Calcium (in milligrams)
9. Iron (in milligrams)
10. Vitamin A (in International Units)
11. Vitamin C (in milligrams)
12. Thiamine (in milligrams)
13. Niacin (in milligrams)
14. Riboflavin (in milligrams)

Ingredients which are specified by volume measurement must have the appropriate conversion factor supplied in field (2). To calculate this factor, find a weight/volume conversion figure for the food item. Multiply this figure by the number of grams per weight unit and divide by the number of milliliters per volume unit. For example, if milk is 8.6 ounces per cup, then:

$$8.6 \text{ oz/cup} \times 28.35 \text{ gm/oz} \div 236.6 \text{ ml/cup} = 1.03 \text{ gm/ml}$$

(Notice that the ounces referred to are weight measurement, not fluid ounces, which are volume.) Weight/volume conversions for many food items can be found in USDA Home & Garden Bulletin No. 72, *Nutritive Value of Foods.* Measurement equivalents are found in many cookbooks. The program can be operated without conversion factors, if all ingredients are specified by weight.

Nutrient amounts in the appropriate units, per 100 grams of food item, can be taken directly from USDA Handbook No. 8, *Composition of Foods.* These numbers can also be computed from other data sources, such as USDA Bulletin 72. Verify that the units are correct (mg, IU, etc.) for all nutrients, or apply conversion factors as necessary. Normalize to 100 grams. (If the data is for 125 grams, divide all numbers by 1.25. If for one pound, divide by 4.536.)

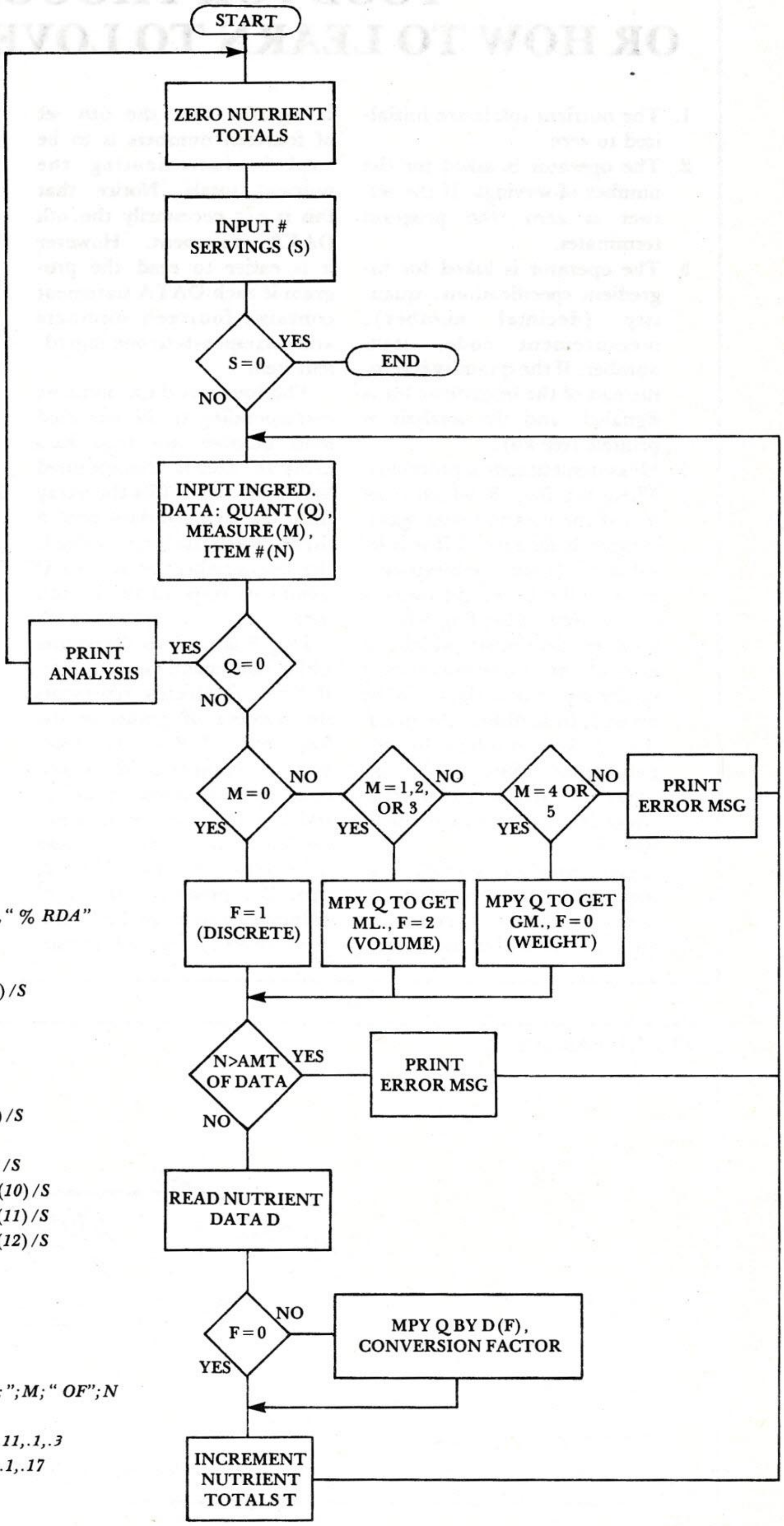
It is essential to have all fourteen values for each ingredient item. Otherwise, there is a risk of analysis errors or program failure due to data pool exhaustion. One way to verify the quantity of data is to run an analysis for 3.527 ounces (100 grams) of the last food item in the data pool. The analysis results should match the data for that item.

Once you have finished adding data you must change statement 400. Substitute for number 3 the new total of ingredients for which the program now has data.

PERSONAL NUTRITIONAL ANALYSIS PROGRAM

```
10 REM NUTRIVALUE I
15 REM PERSONAL NUTRITIONAL ANALYSIS PROGRAM
20 REM K. BROTHERS
25 REM COPYRIGHT 1977, CONSULTUS
30 REM VARIABLES:
40 REM  T() = NUTRIENT TOTALS
50 REM  D() = INDIVIDUAL NUTRIENT ARRAY
55 REM  I,J = INDEXING VARIABLES
60 REM  F = MEASUREMENT TYPE FLAG
65 REM  S = NUMBER OF SERVINGS
70 REM  Q = QUANTITY OF INGREDIENT
80 REM  N = ITEM NUMBER (SEQUENTIALLY) OF INGREDIENT
90 REM  M = MEASUREMENT CODE:
100 REM            0—NO UNIT (DISCRETE)
110 REM            1—CUPS
120 REM            2—TABLESPOONS
130 REM            3—TEASPOONS
140 REM            4—POUNDS
150 REM            5—OUNCES
160 REM
199 DIM D(14),T(12)
200 FOR J=1 TO 12
240 LET T(J) =0
250 NEXT J
```

```
260 PRINT "# SERVINGS";
270 INPUT S
280 IF S=0 THEN 900
290 PRINT "INGREDIENTS: QUANTITY,MEASUREMENT,ITEM NUMBER"
300 RESTORE
302 INPUT Q,M,N
305 IF Q=0 THEN 500
310 IF M=0 THEN 350
315 IF M=1 THEN 360
320 IF M=2 THEN 365
325 IF M=3 THEN 370
330 IF M=4 THEN 385
335 IF M=5 THEN 390
340 PRINT "ILLEGAL MEASUREMENT:";M
345 GOTO 300
350 LET F=1
355 GOTO 400
360 LET Q=Q*16
365 LET Q=Q*3
370 LET Q=Q*4.93
375 LET F=2
380 GOTO 400
385 LET Q=Q*16
390 LET Q=Q*28.35
395 LET F=0
400 IF N 3 THEN 600
405 FOR I=1 TO N
410 FOR J=1 TO 14
415 READ D(J)
420 NEXT J
425 NEXT I
430 IF F=0 THEN 450
435 IF D(F)=0 THEN 610
440 LET Q=Q*D(F)
450 FOR J=1 TO 12
455 LET T(J)=T(J)+Q*D(J+2)/100
460 NEXT J
470 GOTO 300
500 PRINT
505 PRINT
510 PRINT "NUTRIENT"," TOTAL"," PER SVG"," % RDA"
515 PRINT
520 PRINT "CALORIES",T(1),T(1)/S
525 PRINT "PROT(GM)",T(2),T(2)/S,2.222*T(2)/S
530 PRINT "CARB(GM)",T(3),T(3)/S
535 PRINT "FATS(GM)",T(4),T(4)/S
540 PRINT "SODI(MG)",T(5),T(5)/S
545 PRINT "CALC(MG)",T(6),T(6)/S,.1*T(6)/S
550 PRINT "IRON(MG)",T(7),T(7)/S,5.556*T(7)/S
555 PRINT "VITA(IU)",T(8),T(8)/S,.02*T(8)/S
560 PRINT "VITC(MG)",T(9),T(9)/S,1.667*T(9)/S
565 PRINT "THIA(MG)",T(10),T(10)/S,83.33*T(10)/S
570 PRINT "NIAC(MG)",T(11),T(11)/S,6.667*T(11)/S
575 PRINT "RIBO(MG)",T(12),T(12)/S,66.67*T(12)/S
580 PRINT
585 PRINT
590 GOTO 200
600 PRINT "ITEM NUMBER TOO HIGH:";N
605 GOTO 300
610 PRINT "NO MEASUREMENT CONVERSION:";M;" OF";N
615 GOTO 300
700 DATA 50,0,163,12.9,.9.11.5,122,54,2.3,1180,0,.11,.1,.3
710 DATA 0,1.03,65,3.5,4.9,3.5,50,118,0,140,1,.03,.1,.17
715 DATA 0,.85,385,0,96.4,0,1,0,.1,0,0,0,0,0
900 END
```


START
ZERO NUTRIENT TOTALS
INPUT # SERVINGS (S)
S=0
YES
NO
END
INPUT INGRED. DATA: QUANT(Q), MEASURE(M), ITEM #(N)
Q=0
YES
NO
PRINT ANALYSIS
M=0
NO
YES
M=1,2, OR 3
NO
YES
M=4 OR 5
NO
YES
PRINT ERROR MSG
F=1 (DISCRETE)
MPY Q TO GET ML., F=2 (VOLUME)
MPY Q TO GET GM., F=0 (WEIGHT)
N>AMT OF DATA
YES
NO
PRINT ERROR MSG
READ NUTRIENT DATA D
F=0
NO
YES
MPY Q BY D(F), CONVERSION FACTOR
INCREMENT NUTRIENT TOTALS T

FOOD FOR THOUGHT OR HOW TO LEARN TO LOVE NUTRIENTS

1. **The nutrient totals are initialized to zero.**
2. **The operator is asked for the number of servings. If the answer is zero, the program terminates.**
3. **The operator is asked for ingredient specifications: quantity (decimal number), measurement code, item number. If the quantity is zero, the end of the ingredient list is signaled, and the analysis is printed (see #6).**
4. Measurement code is processed. There is a flag, F, which is set to 1 if the measurement specification is discrete, 2 if it is by volume (cups, tablespoons, etc.), or 0 if by weight (ounces or pounds). This flag will be used to determine which, if any, of the conversion factors in the ingredient data will be needed. In addition, the quantity Q is multiplied by the number of milliliters (if volume) or grams (if weight) equivalent to the measurement specified.
5. Ingredient data is read. The ingredient item number *n,* typed in by the operator, is an index into the data pool. It signifies that the nth set of fourteen numbers is to be used in incrementing the nutrient totals. Notice that this is not necessarily the nth DATA statement. However it is easier to read the program if each DATA statement contains fourteen numbers and corresponds to one ingredient item.

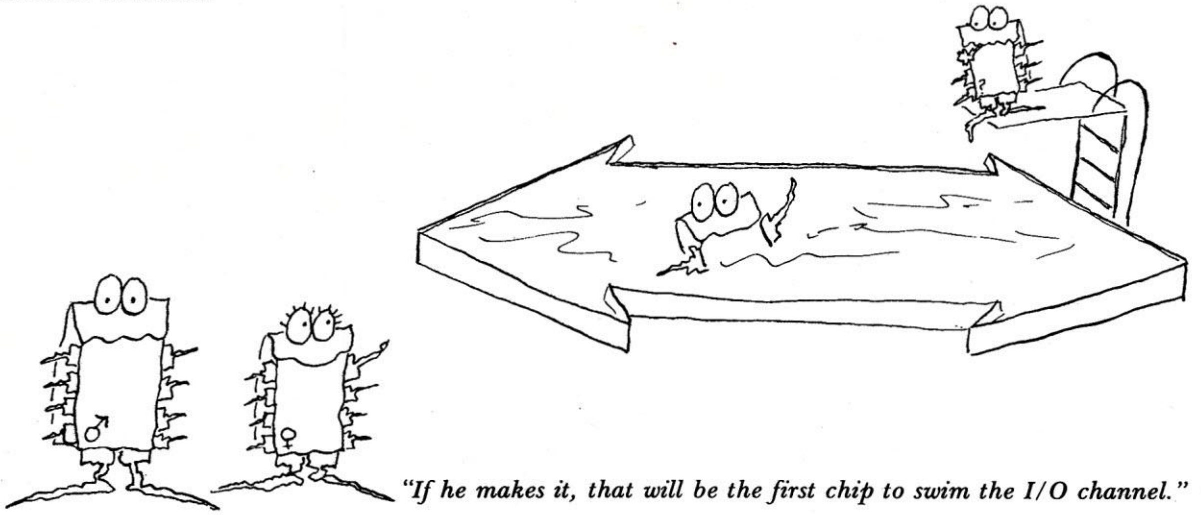
   The fourteen data numbers corresponding to the specified item number are read into array D. This is accomplished by a loop which fills the array from the restored data pool n times. When the loop is exited, the last numbers read into D should correspond to the nth item.

   Flag F is used to determine which conversion factor to use. If F = 0, Q already represents the number of grams of the food item. If F = 1, Q represents the number of items, and must be multiplied by D(1), which is the conversion factor for grams-per-item, to yield the number of grams. If F = 2, then Q represents number of milliliters and must be multiplied by D(2), which is grams-per-ml, to yield number of grams.

   Q, which now is the number of grams of the food item specified, is multiplied by the nutrient data numbers D(3) through D(14) (representing nutrient amount per 100 grams of item), divided by 100 (to get nutrient amount per gram of item), and added to the nutrient totals.

   The program then prompts for additional ingredient specifications (see #3).
6. When the ingredient input is finished, the analysis is printed out. Analysis consists of total amount of each nutrient in the recipe, amount per serving, and (where applicable) percentage of U.S. Recommended Daily Allowance (RDA) per serving. This percentage is obtained by dividing nutrient amount per serving by the RDA and multiplying by 100 (for percent). As implemented, there is a single factor for this computation, equivalent to 100 ÷ RDA.
7. When the output is completed, the program reinitializes for the next recipe.

EVE 'N' PARITY



"If he makes it, that will be the first chip to swim the I/O channel."

Cartoon by Korkie

COME CLOSER

and we won't even have to talk

by Avery Johnson

It's always curious how and where ideas arise because later the instance of their coming may bear a relation to the context of their use. I'd like to tell the story of one such moment because it might make the idea that came out of it clearer.

A few years ago, I was visiting an old friend of mine in Munich; he was just on the verge of leaving the book publishing world to become a TV producer-director. That called for some sort of celebration and I was privileged to be included: an evening-long gourmet dinner at the "Walterspiele" restaurant, one of the finest in the world.

There were about seven of us altogether: three couples from the publishing business and me. The meal was excellent, eaten over hours of lively conversation. That's where both my troubles and my idea began. You must understand that when people who publish books—especially very literary or artistic books—get together, they derive extreme delight from telling each other witty and elegant yarns, all in the most brilliant language possible. But poor me, my German is far from fluent and I haven't much idea what anyone said that evening except when we were examining the menu and deciding what to have. Aha! How about that!? What I started to notice goes as follows.

So long as we were discussing something immediately at hand—on the table, on the menu, in the room—my German was more than adequate for participation. If our consideration shifted to something a little more distant in space or time—something that happened on the way to the restaurant or some item of the day's news—I was able to comprehend fairly well but I couldn't generate much of an intelligent comment.

But when the elegant tales were told and the objects and events of them were either fictitious or at least very far removed from our time and space right there in the restaurant, well, I just couldn't follow any of it. To amuse myself I started listening to the individual words that were tossed back and forth, looking at them as if they were separate objects for study. I made a curious discovery: many of the words were those I could remember having had great trouble with on vocabulary lists in school.

For example, the verbs were those that had complicated prepositional prefixes which greatly (but precisely) modify the meanings of the root words. OK, that follows because it is the prepositions in our language, too, that point to the material, time, and place relationships. Try some: of (*of* wood), over (*over* there), at (*at* noon, *at* my place), and so on. These weren't words that anyone had found it necessary to use while discussing the menu that night in the Munich restaurant, but that's because the stuff of things and their where-and-when are obvious if they are present here, now. If you fabricate a story set in some other time and place, then you have to carry along with you all of the prepositional references so that no one will be confused about what you are referring to.

I returned to this country a day or two later, back to a crazy laboratory on Boston's waterfront that a bunch of us had been working in for almost two years and where we had been experimenting with a variety of non-verbal communication methods. TELEGRASP (see July 1977 *ROM*) was one of the gadgets, but there were lots more. Our main interest was in communicating with people by manipulating the environment, or some part of it, that surrounded them: acoustical changes, "live" furniture, video displays, and so on, but always in such a way that the people were participants rather than passive audience.

Among other efforts, we were looking for a conceptual structure to lead us deeper into the how and why of successful environmental communication. How was it that we could sometimes get so much across with only a grunt or a nudge or a smile combined with a look of the eyes in a particular direction? How was it that at other times or with someone else, a long verbal explanation would be necessary?

I felt I had come back with a piece of the answer. It had to do with some kind of "distance" between the people (or other entities) engaged in a dialogue and the "referent" of that dialogue. The greater the distance, the greater the necessity that the dialogue carry with it the cues as to the where and when and what of the referent. If the dialogue is about the people who are actually having the dialogue and about their relationship to each other (as, for instance, when two people make love), then words aren't necessary at all! As matters move off a bit we can still get along without words pretty well but the areas of ambiguity begin to widen and we must take more care to make our meaning clear.

What we were after in our lab was a way to formalize a theory that would

Diagrams by the author, drawn by Korkie

allow us to make adequate designs for things like automobile seats that would help the driver, auditoriums that would help conferences, theatres that would let the audience enliven a performance, and even manufacturing tools that would let an unskilled operator (the customer?) fashion a shoe or a piece of clothing. In short, we wanted not only a handle on the person-to-person dialogue situation, but also a way to bring to technical fruition a means to make possible "courteous" dialogues between people and their machines.

We wanted a means to make possible "courteous" dialogues between people and their machines.

The usefulness of our research has been greatest for the consideration of man-machine or man-man pairings and it is presented in these contexts. It must therefore be made clear at the outset that no claim is made for the application of these notions to deductive processes: computers programmed in advance to perform ritual acts. The applications that are intended are those in which an ongoing exploration of one system by another is entailed and where no final result nor homeostatic state is anticipated.

Dialogue is a communicative behavior which we have as yet been unable to describe sufficiently well in scientific terms to apply a measure to it. The root of our problem as observers is that if we attempt to identify or separate inputs and outputs to or from the participants, we effectively obliterate the message and render the meaningful exchange empty. We cannot observe the messages because the medium for each participant includes the responsiveness of the other in an infinite recursion.

The isolation and identification of transitory segments of a dialogue is possible, but their meanings when embedded in the broader contexts of the dialogue itself can never fully be known unless one is oneself a participant—whereupon the meanings necessarily take on a self-referent quality knowable only to that knower. The meaning of an object or event to an organism is embodied in his consequent response (real or latent) to it; such is his way of exploring the context of its occurrence. We cannot wholly share that exploration with him. We can, however, describe a dialogue in a qualitative way and can acquire from that description a notion of the complexity of the systems involved and of the interface required to sustain it.

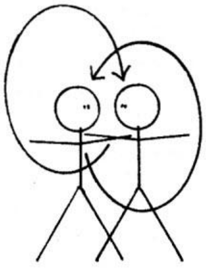
Any communication is *about* something—let us call it the "referent"—even if that something is only the message itself. In general we may say that the referent exists within the information space that may be explored by the participants in the dialogue. Consider the idea that there may be a measure of "distance" in time and/or space between the referent and the moment and place of the dialogue in process. Let us examine the consequences of moving along a scale of this "distance" from zero outward.

Figure 1 provides a graphic and verbal representation of five positions distributed along the scale. The illustrations are more or less familiar instances of interpersonal or human-machine interfacing situations. The positions are numbered 1 through 5 for convenience of discussion, but no implication is intended either that the possible extremes are depicted or that equal spacings of "distance" have been achieved.

Position 1: The referent is the participants themselves and their relation to each other. The communication may be characterized as *immediate* because in fact no media are employed apart from the responsive environment that each of the participants is for the other.

Position 2: We move out to where the referent is external to either participant but where their interactions with it and with each other are indistinguishable. The communication is *communal* and its medium is the referent itself. Examples within the realm of interpersonal communication are common but generally pass unrecognized as dialogue. Blind children, for instance, will not learn to run or jump until they have shared the experience in direct physical contact with another person. Learning to manipulate substances, common shapes, and culturally-established relationships of whole and part is acquired largely through the sharing of toys and other media with adults. More familiar is the way in which heavy objects teach us about gravity, or the way a well-tuned sports car can give the driver a "feel" of the road which no passenger can ever share.

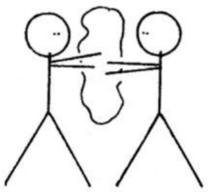
Position 3: Here the participants begin to lose direct contact with each other's reactions and they communicate more by effects which both produce upon a common environment, although the effects themselves may be somewhat remote in space or delayed in their responsiveness. The referent is distinctly separate from the participants and their sharing of it might be termed *adjacent*.



1

LOVE-MAKING
MOTHER & CHILD
PHYSIOTHERAPY

IMMEDIATE



2

SHARED TOYS
"HANDS-ON
INSTRUCTION"
DUAL CONTROLS

COMMUNAL

Note, as we move from one position to the next, a number of other changes that are taking place. Direct reaction back from the medium becomes less important (e.g., a squeeze back from the hand that has been grasped, light switches whose resistance to being moved is not relevant to the status of the lights, on or off states). Instead, the feedback loops upon which a communicant depends in order to monitor the effects produced become increasingly more reliant upon that part of the sensorium which we learn to use passively or at a distance: sight and hearing.

For the human participant there is a decreasing involvement of large-muscle movement: his efferent activity moves toward the fingertips and thence to lips and sign language where direct contact is unnecessary.

As for the communicant's problem of exploring the context of a respondent's messages, the need for more time of exploration and more storage of past experience for reference increases. Ambiguities become more difficult to resolve as they find expression simultaneously in more sensorimotor modalities. Stated another way: expressions of intention are more equivocal because the recipient is less and less a part of the intender's self-referent loops in his environment.

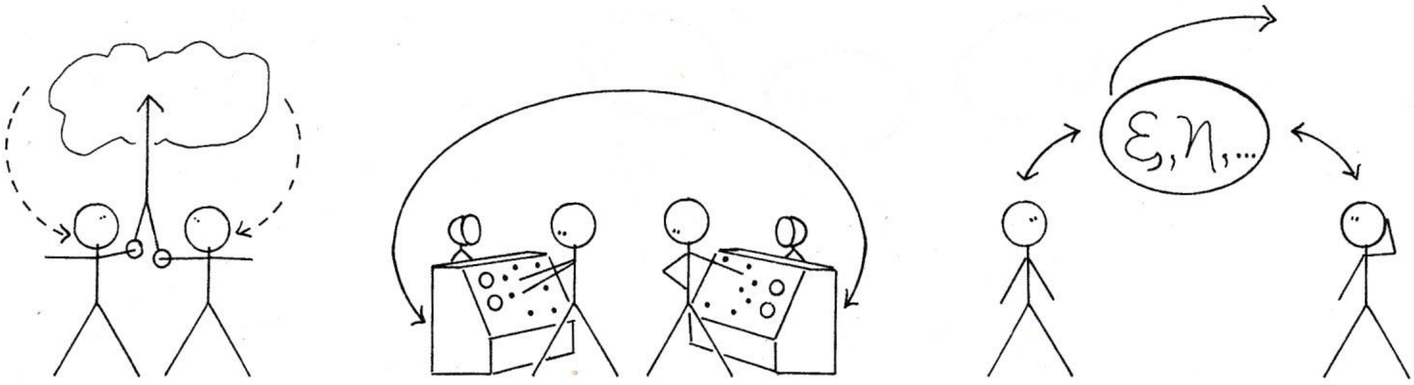


Figure 1
Communication about a Receding Referent

Position 4: At this distance the communication about the referent is *metaphorical* in that its representation is only mapped onto the environment of each participant by implication. The model of the referent is not itself of real substance, as it was in the preceding positions, in the sense that it could be manipulated directly. Rather, the interaction must be brought about by an intervening agent. Chalk on a blackboard, a knob, a pushbutton, or a light-pen on a computer do not react back upon the user except to his eyes or ears. His participation, his involvement in real events, has been reduced to a level where he can no longer have confidence in the reality of the source. An example would be the mayor who no longer really knows how the statistics, gathered by others as a base for his decisions, were obtained; he does not participate in the intentions of the fact-finders.

Position 5: At the furthest distance shown we encounter the familiar world of *symbolic* language and its referents which need not even exist in fact. This is the realm of the telegraph, high-speed printer, speechmaking, books, contract-writing, etc., where the burden of the entailment of the intended meaning of a message rests almost entirely upon the sender's mastery of the medium. I say "almost" because in many cases the transaction between participants can allow for some clarifying questions and answers.

> ***Blind children will not learn to run or jump until they have shared the experience in direct physical contact with others.***

Yet further, a sixth position on the scale might show a referent at the level of metalanguage—of which this article itself may be an example. But Figure 1 does not continue the scale that far.

This proposed scale is intended to allow us to place in rank order the varieties of dialogue which we shall consider. The scale parameter has been referred to simply as distance in time and/or space, and although it may in fact be precisely quantifiable in many cases, for our purposes here it is quite sufficient to be able to specify a comparative ordering of a few examples. We will focus here on broader speculation about what other inferences may be made about a system and the dialogue in which it is engaged, once its predominant position on the scale is known. We have been able to extract a number of valuable design criteria from them.

It should be emphasized again that the communications of interest here are those characteristic of dialogue and *not* those in which information flows in one direction only. If a system is intended to be employed simply as a filter and does not allow the recipient an opportunity to be environment to the source, then a dialogue is not in process. The processes of dialogue are really fundamentally different from

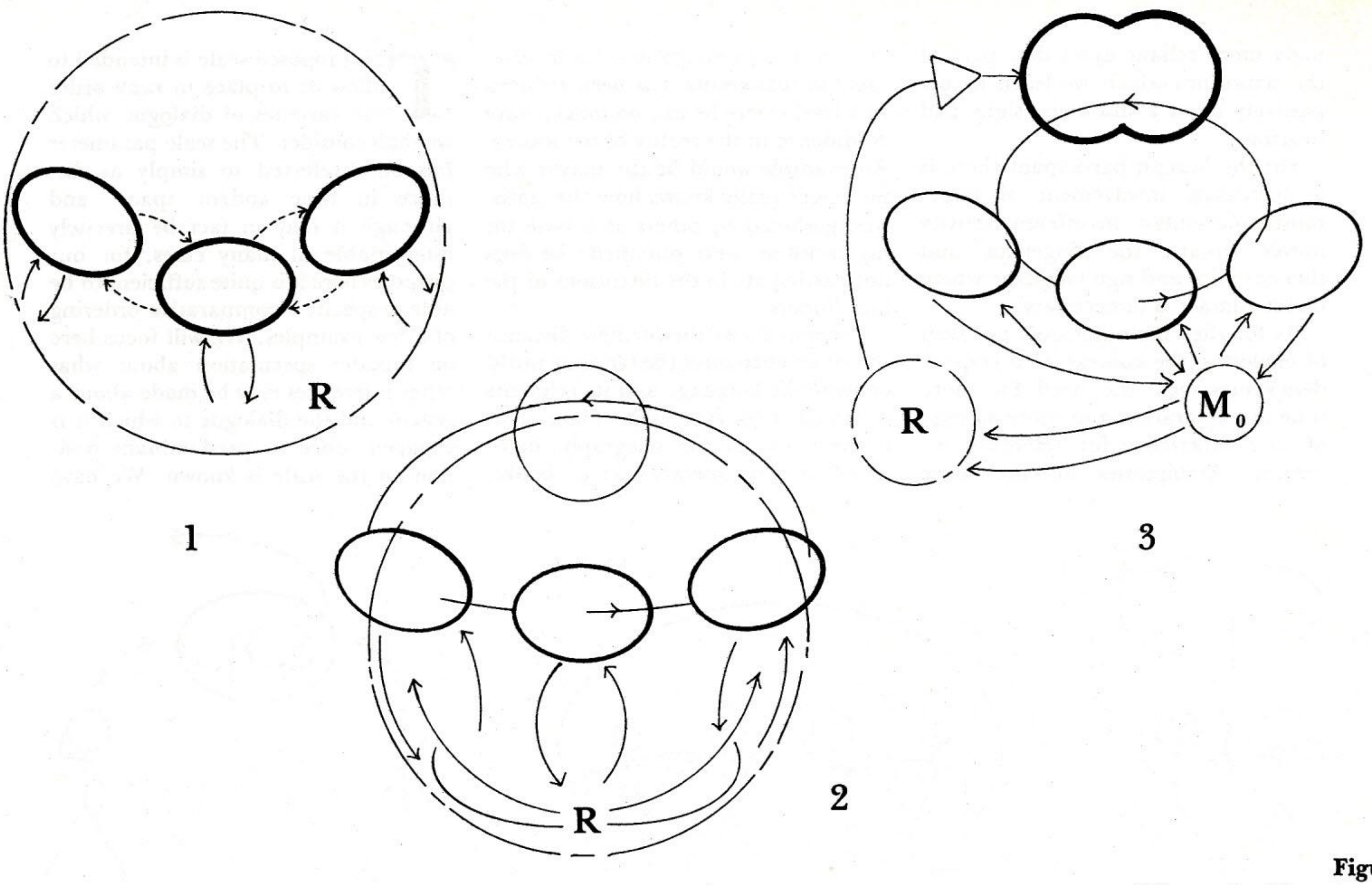


Figure
Schematic Diagrams of

the more familiar "scientific" methods of telling or being told.

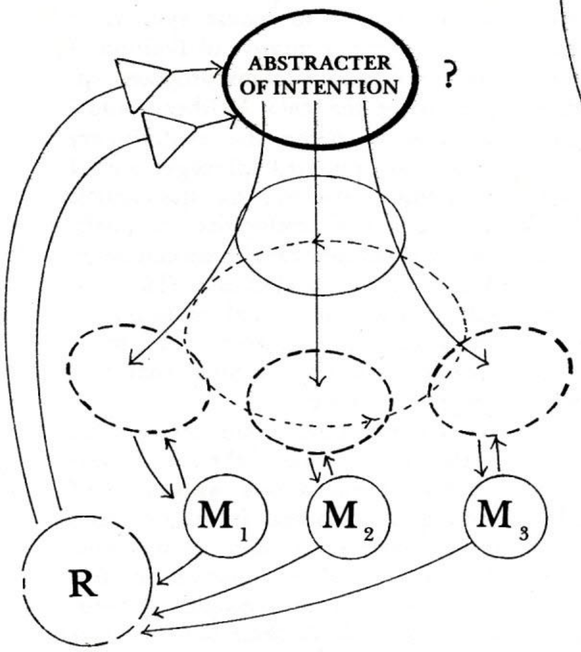
Consider presenting to an engineer, trained in today's state-of-the-art, the problem either of a communication device to employ a participant's tactile sense, or of a prosthetic hand for an amputee. Our best guess is that he will proceed immediately to think in terms of tailoring an input or an output characteristic as if either could valuably exist in isolation. He will present tactile stimuli *seriatim* as patterns to a presumed passive recipient, making quite inappropriate use of a touching-grasping system which depends heavily upon exploration to achieve its highest-resolution "sensing." Dialogue design is needed. As for prosthetic hands, they have not yet been granted self-referent behaviors of their own and are directed as simple inert tools, presenting feedback only to the user's eyes. Neither engineered solution will be suited to the proposed scale because the referents are not describable in terms of the behavior of the systems selected to mediate them: these behaviors have in fact been made irrelevant.

> *The processes of dialogue are fundamentally different from the scientific methods of telling or being told.*

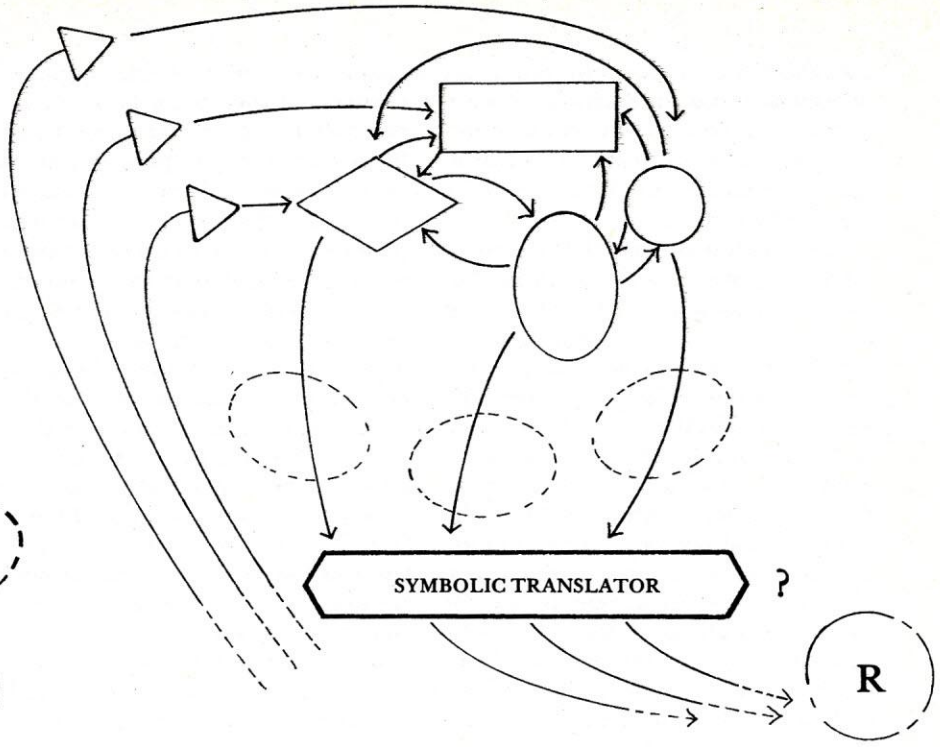
Suppose that one wished to create a purely mechanical device to serve as one of the participants in a dialogue. Until now we have concerned ourselves primarily with relatively immediate or mediated dialogue between persons as the ultimate originators of the information flows. Now we ask whether it might be possible, say, to provide a computer or some other complex system with an interface adapted to the kinds of informal physiological transactions at which human beings are so skilled? Would such a device require a tremendous amount of centralized computation in order to orchestrate its behavior, or relatively little if one organizes it properly with *decentralized* foci of simple, reflexive units?

Figure 2 is an attempt at a simple schematic view of the increasing levels of system complexity that are required as one moves up the scale: that is, as the perceiver's task of modeling the referent becomes correspondingly more complex. Each diagram (numbered to correspond to the same respective scale positions as in Figure 1) is intended to depict only the mechanical participant in the dialogue and to summarize graphically the relationships of the parts of the interfacing mechanism to each other and to the referent.

In each diagram the referent of the dialogue is depicted as a dashed circle and is labeled *R*. It will be seen in Diagram 1 to surround the rest of the figure, indicating that the referent is the dialogue itself and the participants in it; in Diagram 5 the referent is suitably separate and remote. The small ellipses represent separate active components of



4



5

M_k = MEDIUM OF COMMUNICATION

R = REFERENT

the physical interface. These might be air bags in a piece of furniture or water jets in a hydrotherapy bath. In each diagram only three are shown, but in a real device there would more likely be dozens or hundreds of them. Each has a means of "local computation" enabling it to respond energetically to changes in its immediate environment.

The diagrams imply that as one moves up the scale the significance of the local computations diminishes and a means of central organization acquires control. By the time we reach Diagram 5 the incoming information is being brought directly to the central controller: input and output no longer share a common perceptual space.

Skills that human beings develop easily are being made irrelevant by machines which cannot recognize our messages.

Let us return to a more philosophical level of argument inasmuch as the diagrams are intended only to suggest a general aspect of connectedness of parts. Each participant in the dialogue is attempting to explore a referent to determine the intention conveyed upon it by the other. Let me rephrase the foregoing sentence so that its meaning will not be obscure, for again the exceptional properties of dialogue are such that its triadic nature is not revealed by our more familiar modes of description. If the referent were itself inanimate and were to be explored by a single observer, then there would arise no difficulty on the part of a second, passive observer, (wire tapper, eavesdropper) in bearing witness to the messages. However, if a dialogue is in process, then the respondent is imposing frames of reference upon the referent which it did not possess before and which, for their description, partake of the participation itself. One of the problems upon which this article is intended to shed light is that of placing some kind of upper bound on the complexity actually required for a realizable system engaged in a dialogue at various levels of the proposed scale.

Most of the work in which the author and his colleagues have been engaged is focused on systems to be found at the lower end of the scale, Position 1. Suppose, for example, one were to fashion a hospital bed which would participate with the patient in manipulating him for orthopedic therapy; or suppose that an automobile seat were to be made that would concern itself with the comfort and alertness of the driver (i.e., the referent is the relationship of, between, and within the participants). It will be found that one needs no more complexity of computation within the mechanism than a simple reflexive responsiveness of each component to its own local environment, and that this may in fact be provided within the structure of the components themselves. That is, as the physical relationships of the aggregate change, they thereby alter their own properties of responsiveness. Interactions between elements of the interface will be effective and need not be explicitly controlled as separate computations by a "remote" computer.

We have found that the liveliness of interaction of such a system with a human respondent is greatly enhanced if the interfacing mechanism is sufficiently sensitive to its own changes that it will sustain a slow, quasi-periodic movement of its own: not so slow as to be undiscernible (below respiration rates) but not so fast as to be alarming (above cardiac rhythms). While the device should not be preprogrammed to move, it should be mildly active even when the respondent is not. Of course, variations upon this self-generated activity produce a system which is even more useful. It is just such a self-adjusting variability that the next position offers.

Moving along to Position 2 of the scale, one might want to provide a means of "holding hands at a distance" (we have proposed a device called "TELEGRASP") or of communicating to a driver some of the conditions of his near surroundings (see Table). The necessity arises for computations among the components of the interfacing system which give them a behavior that is purposive in the aggregate. That is, the longer time-grain and space-grain of its communications require that the organizing elements of the system be sensitive to the correlation of responses that are further separated in space or time than were those of Position 1. In our own research work it appears that multi-parameter, self-organizing controllers will suffice, and that their computations of performance assessment may be carried out in terms of the behavior of the component modules themselves; no additional sensorium is needed.

Further progression up the scale enriches the complexity of system realization. The immediate interactions between the components of the interface and their environment become less and less useful, but this is not to say that those interactions may be obviated for system function. Rather, the description of a receding referent becomes less relevant to the detailed interactions at the interface. The performance-assessment criteria of the system as a whole must be established by way of a more external, indirect means of observation. Figure 1 and Figure 2 show a similar progression. Each higher position should be viewed as an elaboration upon the one below it.

Positions 4 and 5 correspond to ways in which computers are familiarly employed today. Computer graphics in architecture is a recent example of human-machine "metaphorical" dialogue. Computer programming via compiler illustrates a symbolic exchange at a teletype.

Table of Some Analogical Examples

Four examples of application of the scale to different descriptive problems. The first three should be understandable in the light of arguments on dialogue and referents. The fourth is an example of the properties of an adequate interface.

	COURTSHIP	ARCHITECTURE	CHILD DEVELOPMENT	ANIMATED AUTOMOBILE SEAT TO COMMUNICATE WITH DRIVER
1	Making love	Living in the space himself	Infant body-contact	Comfort and alertness
2	Holding hands	Carpentry	Sharing toys	Conditions of auto
3	Walking together	Construction foreman	Shared environment	"Feel" of the road
4	Videophone	Drafting	Mimicry	Map follower
5	Writing letters	Verbal specifications	Speech	Road sign symbol monitor

Direction of increasing system complexity and decreasing involvement

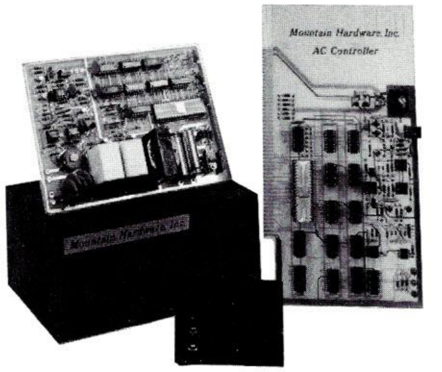
Let me state unequivocally that I do *not* know how to realize systems of hardware representative of Positions 4 and 5 if I go at it by my proposed approach: *up* the scale. Neither would I presume to name the evolutionary process so embodied "phylogenetic." I do predict, however, that the current interest in the development of small, locally-dedicated computers can benefit from these considerations. They will achieve their individual performance levels through a process of "ontogenetic" learning rather than by preprogramming.

There is without doubt much value in the current state-of-the-art of computers; tremendous amounts of thought and capital have gone into their elaboration and refinement. They fall far short, however, of fulfilling a need for complex systems which can provide their skills to users who themselves have only informal means of communication: children in particular, but in addition a vast population of potential users who have tasks they want performed for which no algorithms exist simply because in any instance the user has not yet "decided" what it is he wants. At present, the kinds of skills that human beings develop easily are being made irrelevant by machines which cannot recognize our messages nor elicit our intentions.

The major population for which we would like to see such efforts directed is that of the world's children from age zero (literally) onward. At no other time in history has it been so difficult for a child to raise the information level of his own environment to a point where it responds to him in his style. The computer is looked upon askance by most young students as just one more addition to an educational process designed to make them look stupid. My own contention is that this opinion can be reversed but that it will take too long to achieve systems of necessary dialogue design if one works down the scale from the top. There is the further, real danger that the embodiments arrived at would be far more complicated than necessary.

All along the scale I have suggested there are simplicities to be exploited; it is my hope that the scale itself will help to identify them. ▼

IN CONTROL WITH INTROL



Control, control, everyone wants power to control the world—or at least his house. Now, with Mountain Hardware's S-100 compatible INTROL system, you can have it. Its AC CONTROLLER board transmits and receives signals over your normal in-house 110 VAC lines, using a wall socket plug-in AC interface adaptor. AC REMOTE units may be plugged in at any location to control electrical devices and appliances. Operates and continuously monitors up to sixty-four remote electrical stations.

The AC CONTROLLER is available in kit form for $149, the AC REMOTE for $99.

Mountain Hardware, Inc.
P. O. Box 1133
Ben Lomond, CA 95005
(408) 336-2495

??????????????????????????????
???????????

dilithium PRESS
HOME COMPUTERS: 2¹⁰ Questions & Answers Volume 2: Software
Rich Didday
dilithium PRESS
HOME COMPUTERS: 2¹⁰ Questions & Answers Volume 1: Hardware
Rich Didday

dilithium Press
P. O. Box 92
Forest Grove, OR 97116

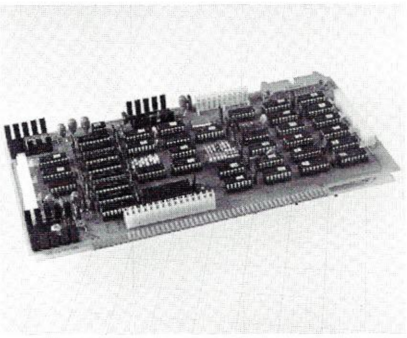
What the world needs is a good basic beginner's text on microcomputers. Though this isn't it, *Home Computers: 2^{10} Questions and Answers* is a step in the right direction. Although the long-delayed publication of this two-volume set shows up in its out-of-date bibliography as well as the computers omitted, and the index is very sparse, it's still something that belongs on every beginner's bookshelf. The partially tape-recorded folksy question-and-answer approach is very interesting, though from an organizational point of view it can drive the average reader crazy. From dilithium Press, $14.90 the set.

Z-TEL FOR XITAN SPELLS IT RIGHT FOR YOU

Everyone talks about text-editing systems, but hardly anyone does anything about them. Now Technical Design Labs does. Z-TEL is here for editing and manipulating your text files. No more deletions or manual text retyping. Cassette $40, paper tape $50.

And to help display your fancy fingerworks there's TDL's new S-100 VDB. A piggyback board utilizing one edge-connector but occupying two board spaces, the VDB uses its own display buffer, leaving your computer memory untouched and available for user programs.

The buffer provides two pages of display, with twenty-five rows of eighty characters each per page. Full ninety-six upper and lower ASCII characters with descenders, plus sixty-four additional display symbols. Hardware blinking cursor, character blink, and lots of other goodies—$349 in kit form, $449 fully assembled and tested.

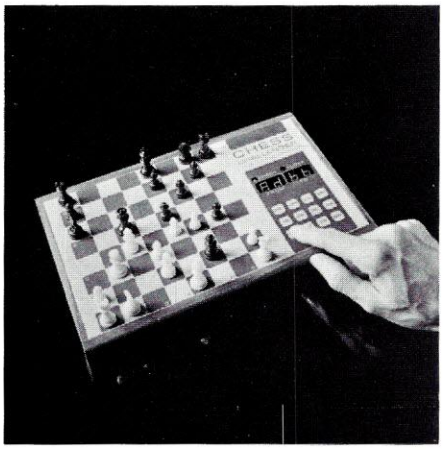


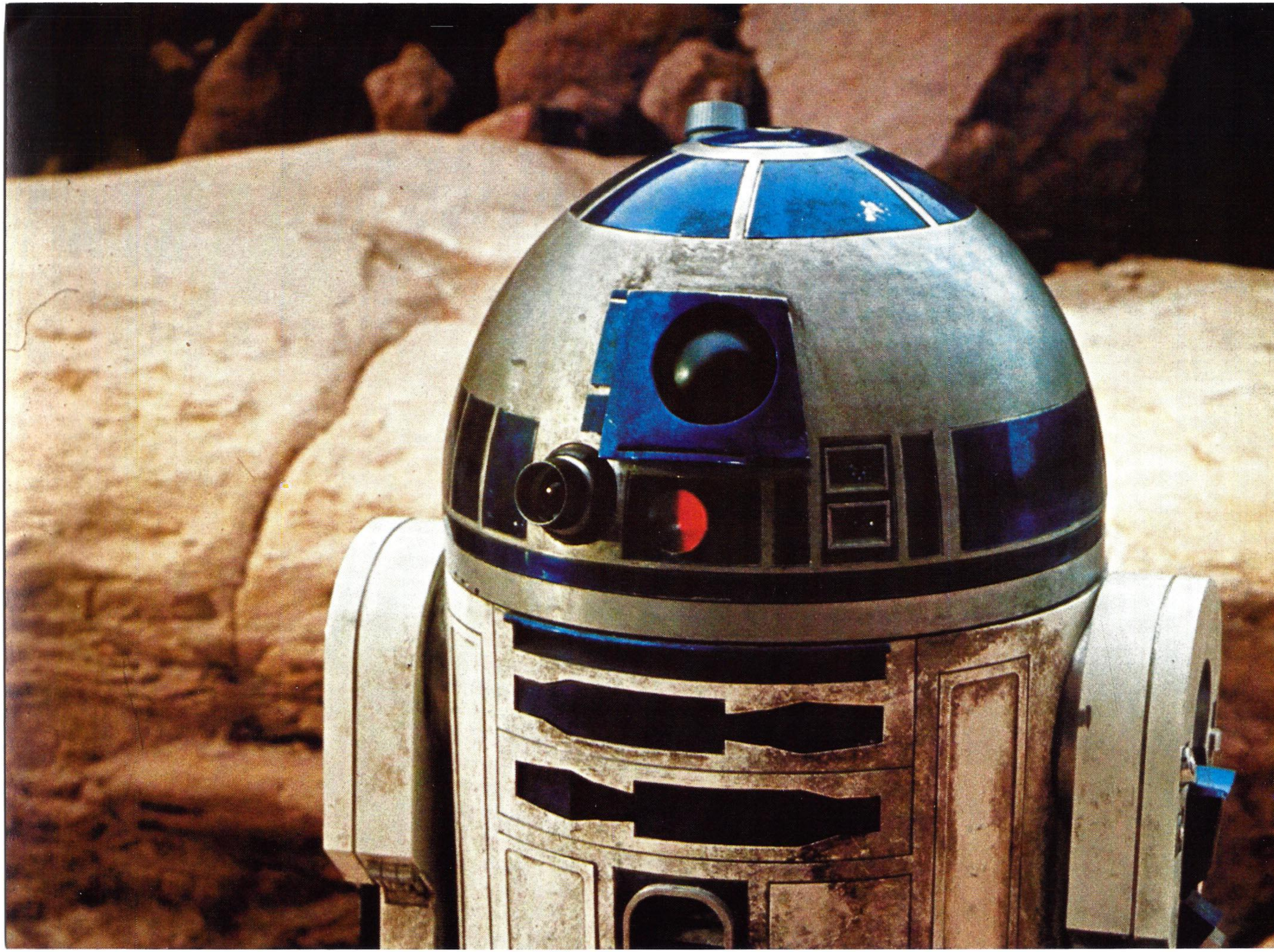
Technical Design Labs
Research Park Bldg H.
1101 State Road
Princeton, NJ 08540
(609) 921-0321

CHECKMATE

Feel like playing chess with someone, or rather something, that won't argue about moves? Try Chess Challenger, a dedicated computer from Fidelity Electronics. And if you begin beating it too often, just send the Challenger back; they'll make life a little harder for you.

Fidelity Electronics Ltd.
5245 Diversey Ave.
Chicago, IL 60639
(312) 237-8090





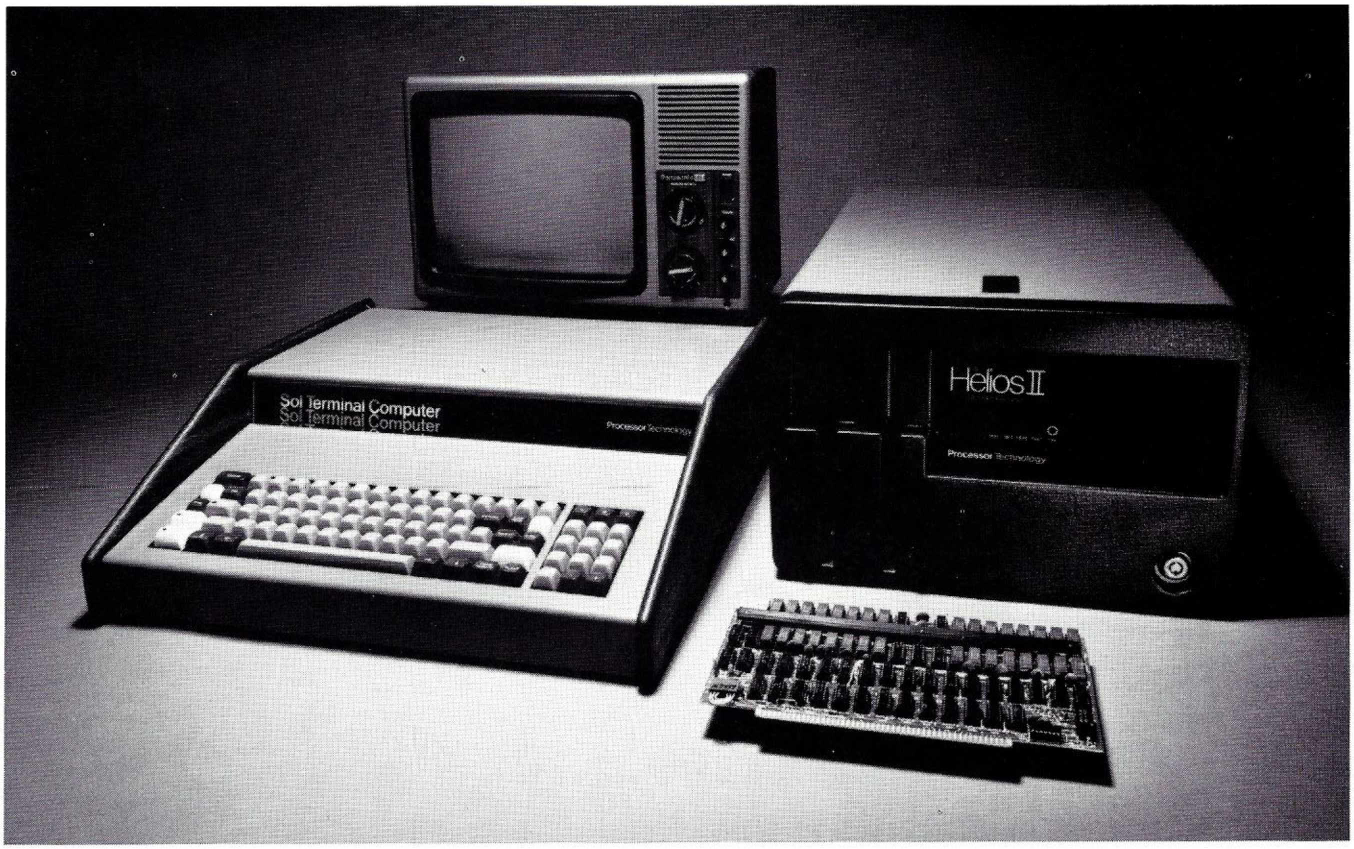
December 1977

ROM
COMPUTER APPLICATIONS FOR LIVING

This month's **ROM** centerfold is the redoubtable R2-D2, considered by many Star Wars' real hero. Asked what lay in the future, now that fame had come his way, R2-D2 replied with a mysterious smile, "Only my programmer knows for sure."

Photograph courtesy of The Star Wars Corp.

© 1977 Twentieth Century-Fox Corp. All Rights Reserved.



Sol-20. First it was THE SMALL COMPUTER. Now, it's THE SMALL COMPUTER SYSTEM.

A year ago, we introduced the Sol-20. It wasn't the first small computer. It was the first complete small computer with everything needed to get it up and on the air as it came from the factory. The keyboard, interfaces, extra memory, factory backup, and service notes were all there.

The results are in: Sol-20 is now the number one small computer in the world. Sols aren't the cheapest, just the most valuable.

We originally designed the Sol-20 as the heart of a complete computer system. So now to solve the problems of science, engineering, education, business management and control and manufacturing, we offer fixed price Sol systems in either kit or fully tested and assembled form. We offer language flexibility, Extended BASIC, ASSEMBLER, PILOT BASIC and FORTRAN IV. We offer Helios II/PTDOS, an extraordinarily capable disk operating system. And remember, though we call these small or personal computer systems, they have more power per dollar than anything ever offered. They provide performance fully comparable and often superior to mini-computer systems costing tens of thousands of dollars more.

What you get. What it costs.

Typical systems include Sol System I priced at $1649 in kit form, $2129 fully assembled and tested. Included are a Sol-20/8 with SOLOS personality module storing essential system software, an 8192 word memory, a 12″ TV/video monitor, a cassette recorder with BASIC tape and all necessary cables.

Sol System II has the same equipment with a larger capacity 16,384 word memory. It sells for $1883 in kit form; $2283 fully assembled.

For even more demanding tasks, Sol System III features Sol-20/16 with SOLOS, 32,768 words of memory, the video monitor and the dual drive Helios II Disk Memory System with the PTDOS disk operating system and Extended DISK BASIC Diskette. Prices, $4750 in kit form, $5450 fully assembled and tested.

More information.

For the most recent literature and a demonstration, see your dealer listed below. Or if more convenient, contact us directly. Please address Processor Technology Corporation, Box O, 7100 Johnson Industrial Drive, Pleasanton, CA 94566. Phone (415) 829-2600.

Processor Technology

AZ: Tempe (602)894-1129; Phoenix (602)942-7300; Tucson (602)327-4579. CA: Berkeley (415)845-6366; Costa Mesa (714)646-0221; Fresno (209)266-9566; Hayward (415)537-2983; Lawndale (213)371-2421; Orange (714)633-1222; Pasadena (213)684-3311; Sacramento (916)443-4944; San Francisco (415)431-0640, (415)421-8686; San Jose (408)377-4685, (408)226-8383; San Rafael (415)457-9311; Santa Clara (408)249-4221; Sunnyvale (408)735-7480; Tarzana (213)343-3919; Van Nuys (213) 786-7411; Walnut Creek (415)933-6252; Westminster (714)894-9131. CO: Boulder (303)449-6233; Englewood (303)761-6232. FL: Fort Lauderdale (305)561-2983; Miami (305)264-2983; Tampa (813)879-4301. GA: Atlanta (404)455-0647. IL: Champaign (217)359-5883; Evanston (312)328-6800; Lombard (312)620-5808. IN: Bloomington (812)334-3607; Indianapolis (317)842-2983, (317)251-3139. IA: Davenport (319)386-3330. KY: Louisville (502)456-5242. MI: Ann Arbor (313)995-7616; Royal Oak (313)576-0900; Troy (313)362-0022. MN: Minneapolis (612)927-5601. NJ: Hoboken (201)420-1644; Iselin (201)283-0600. NY: Middle Island (516)732-4446; New York City (212)686-7923; White Plains (914)949-3282. NC: Raleigh (919)781-0003. OH: Columbus (614)486-7761; Dayton (513)296-1248. OR: Beaverton (503)644-2686; Eugene (503)484-1040; Portland (503)223-3496. RI: Warwick (401)738-4477. SC: Columbia (803)771-7824. TN: Kingsport (615)245-8081. TX: Arlington (817)469-1502; Houston (713)526-3456, (713)772-5257; Lubbock (806)797-1468; Richardson (214)231-1096. VA: McLean (703)821-8333; Reston (703)471-9330; Virginia Beach (804)340-1977. WA: Bellevue (206)746-0651; Seattle (206)524-4101. WI: Milwaukee (414)259-9140. WASHINGTON D.C.: (203)362-2127. CANADA: Ottawa (613)236-7767; Toronto (416)484-9708, (416)482-8080, (416)598-0262; Vancouver (604)736-7474, (604)438-3282.

See Sol here...

ARIZONA

Byte Shop Tempe
813 N. Scottsdale Rd.
Tempe, AZ 85281
(602) 894-1129

Byte Shop Phoenix
12654 N. 28th Dr.
Phoenix, AZ 85029
(602) 942-7300

Byte Shop Tucson
2612 E. Broadway
Tucson, AZ 85716
(602) 327-4579

CALIFORNIA

The Byte Shop
1514 University Ave.
Berkeley, CA 94703
(415) 845-6366

Computer Center
1913 Harbor Blvd.
Costa Mesa, CA 92627
(714) 646-0221

DCI Computer Systems
4670 N. El Capitan
Fresno, CA 93711
(209) 266-9566

The Byte Shop
1122 "B" Street
Hayward, CA 94541
(415) 537-2983

The Byte Shop
16508 Hawthorne Blvd.
Lawndale, CA 90260
(213) 371-2421

The Computer Mart
633-B West Katella
Orange, CA 92667
(714) 633-1222

Byte Shop
496 South Lake Ave.
Pasadena, CA 91101
(213) 684-3311

Micro-Computer
Application Systems
2322 Capitol Avenue
Sacramento, CA 95816
(916) 443-4944

The Computer Store
of San Francisco
1093 Mission Street
San Francisco, CA 94103
(415) 431-0640

Byte Shop
321 Pacific Ave.
San Francisco, CA 94111
(415) 421-8686

The Byte Shop
2626 Union Avenue
San Jose, CA 95124
(408) 377-4685

The Computer Room
124H Blossom Hill Rd.
San Jose, CA 95123
(408) 226-8383

The Byte Shop
509 Francisco Blvd.
San Rafael, CA 94901
(415) 457-9311

The Byte Shop
3400 El Camino Real
Santa Clara, CA 95051
(408) 249-4221

Recreational Computer
Centers
1324 South Mary Ave.
Sunnyvale, CA 94087
(408) 735-7480

Byte Shop of Tarzana
18423 Ventura Blvd.
Tarzana, CA 91356
(213) 343-3919

Computer Components
5848 Sepulveda Blvd.
Van Nuys, CA 91411
(213) 786-7411

The Byte Shop
2989 North Main St.
Walnut Creek, CA 94596
(415) 933-6252

Byte Shop
14300 Beach Blvd.
Westminster, CA 92683
(714) 894-9131

COLORADO

Byte Shop
3101 Walnut St.
Boulder, CO 80301
(303) 449-6233

Byte Shop
3464 S. Acoma St.
Englewood, CO 80110
(303) 761-6232

FLORIDA

Byte Shop of
Fort Lauderdale
1044 East Oakland Park
Blvd.
Ft. Lauderdale, FL 33334
(305) 561-2983

Byte Shop of Miami
7825 Bird Road
Miami, FL 33155
(305) 264-2983

Microcomputer
Systems Inc.
144 So. Dale Mabry Hwy.
Tampa, FL 33609
(813) 879-4301

GEORGIA

Atlanta Computer Mart
5091-B Buford Hwy.
Atlanta, GA 30340
(404) 455-0647

ILLINOIS

Champaign Computer
Company
318 N. Neil Street
Champaign, IL 61820
(217) 359-5883

itty bitty machine co.
1322 Chicago Ave.
Evanston, IL 60201
(312) 328-6800

itty bitty machine co.
42 West Roosevelt
Lombard, IL 60148
(312) 620-5808

INDIANA

The Data Domain
406 So. College Ave.
Bloomington, IN 47401
(812) 334-3607

The Byte Shop
5947 East 82nd St.
Indianapolis, IN 46250
(317) 842-2983

The Data Domain
7027 N. Michigan Rd.
Indianapolis, IN 46268
(317) 251-3139

IOWA

The Computer Store
of Davenport
4128 Brady Street
Davenport, IA 52806
(319) 386-3330

KENTUCKY

The Data Domain
3028 Hunsinger Lane
Louisville, KY 40220
(502) 456-5242

MICHIGAN

The Computer Store
of Ann Arbor
310 East Washington
Ann Arbor, MI 48104
(313) 995-7616

Computer Mart
of Royal Oak
1800 W. 14 Mile Rd.
Royal Oak, MI 48073
(313) 576-0900

General Computer Store
2011 Livernois
Troy, MI 48084
(313) 362-0022

MINNESOTA

Computer Depot, Inc.
3515 W. 70th St.
Minneapolis, MN 55435
(612) 927-5601

NEW JERSEY

Hoboken Computer Works
No. 20 Hudson Place
Hoboken, NJ 07030
(201) 420-1644

The Computer Mart
of New Jersey
501 Route 27
Iselin, NJ 08830
(201) 283-0600

NEW YORK

The Computer Shoppe
444 Middle Country Rd.
Middle Island, NY 11953
(516) 732-4446

The Computer Mart
of New York
118 Madison Ave.
New York, NY 10001
(212) 686-7923

The Computer Corner
200 Hamilton Ave.
White Plains, NY 10601
(914) 949-3282

NORTH CAROLINA

ROMs 'N' RAMs
Crabtree Valley Mall
Raleigh, NC 27604
(919) 781-0003

OHIO

Byte Shop
2432 Chester Lane
Columbus, OH 43321
(614) 486-7761

Computer Mart of Dayton
2665 S. Dixie Ave.
Dayton, OH 45409
(513) 296-1248

OREGON

Byte Shop Computer Store
3482 SW Cedar Hills Blvd.
Beaverton, OR 97005
(503) 644-2686

The Real Oregon
Computer Co.
205 West 10th Ave.
Eugene, OR 97401
(503) 484-1040

Byte Shop Computer Store
2033 SW 4th Ave.
Portland, OR 97201
(503) 223-3496

RHODE ISLAND

Computer Power, Inc.
M24 Airport Mall
1800 Post Rd.
Warwick, RI 02886
(401) 738-4477

SOUTH CAROLINA

Byte Shop
2018 Green St.
Columbia, SC 29205
(803) 771-7824

TENNESSEE

Microproducts & Systems
2307 E. Center Street
Kingsport, TN 37664
(615) 245-8081

TEXAS

Computer Port
926 N. Collins
Arlington, TX 76011
(817) 469-1502

Computertex
2300 Richmond Ave.
Houston, TX 77006
(713) 526-3456

Interactive Computers
7646½ Dashwood Rd.
Houston, TX 77036
(713) 772-5257

Neighborhood Computer
Store
#20 Terrace
Shopping Center
4902 - 34th Street
Lubbock, TX 79410
(806) 797-1468

The Micro Store
634 So. Central
Expressway
Richardson, TX 75080
(214) 231-1096

VIRGINIA

The Computer Systems
Store
1984 Chain Bridge Rd.
McLean, VA 22101
(703) 821-8333

Media Reactions Inc.
Reston International Center
11800 Sunrise Valley Dr.
Suite #312
Reston, VA 22091
(703) 471-9330

The Home Computer Center
2927 Virginia Beach Blvd.
Virginia Beach, VA 23452
(804) 340-1977

WASHINGTON

Byte Shop Computer Store
14701 N.E. 20th Ave.
Bellevue, WA 98007
(206) 746-0651

The Retail Computer Store
410 N.E. 72nd
Seattle, WA 98115
(206) 524-4101

WISCONSIN

The Milwaukee
Computer Store
6916 W. North Ave.
Milwaukee, WI 53213
(414) 259-9140

WASHINGTON D.C.

Georgetown
Computer Store
3286 M Street NW
Washington, D.C. 20004
(203) 362-2127

CANADA

Trintronics
160 Elgin St.
Place Bell Canada
Ottawa, Ontario K2P 2C4
(613) 236-7767

Computer Mart Ltd.
1543 Bayview Ave.
Toronto, Ontario M1K 4K4
(416) 484-9708

First Canadian
Computer Store Ltd.
44 Eglinton Ave. West
Toronto, Ontario M4R 1A1
(416) 482-8080

The Computer Place
186 Queen St. West
Toronto, Ontario M5V 1Z1
(416) 598-0262

Basic Computer Group Ltd.
1548 East 8th Ave.
Vancouver, B.C. V6J 4R8
(604) 736-7474

Pacific Computer Store
4509 Rupert St.
Vancouver, B.C. V5R 2J4
(604) 438-3282

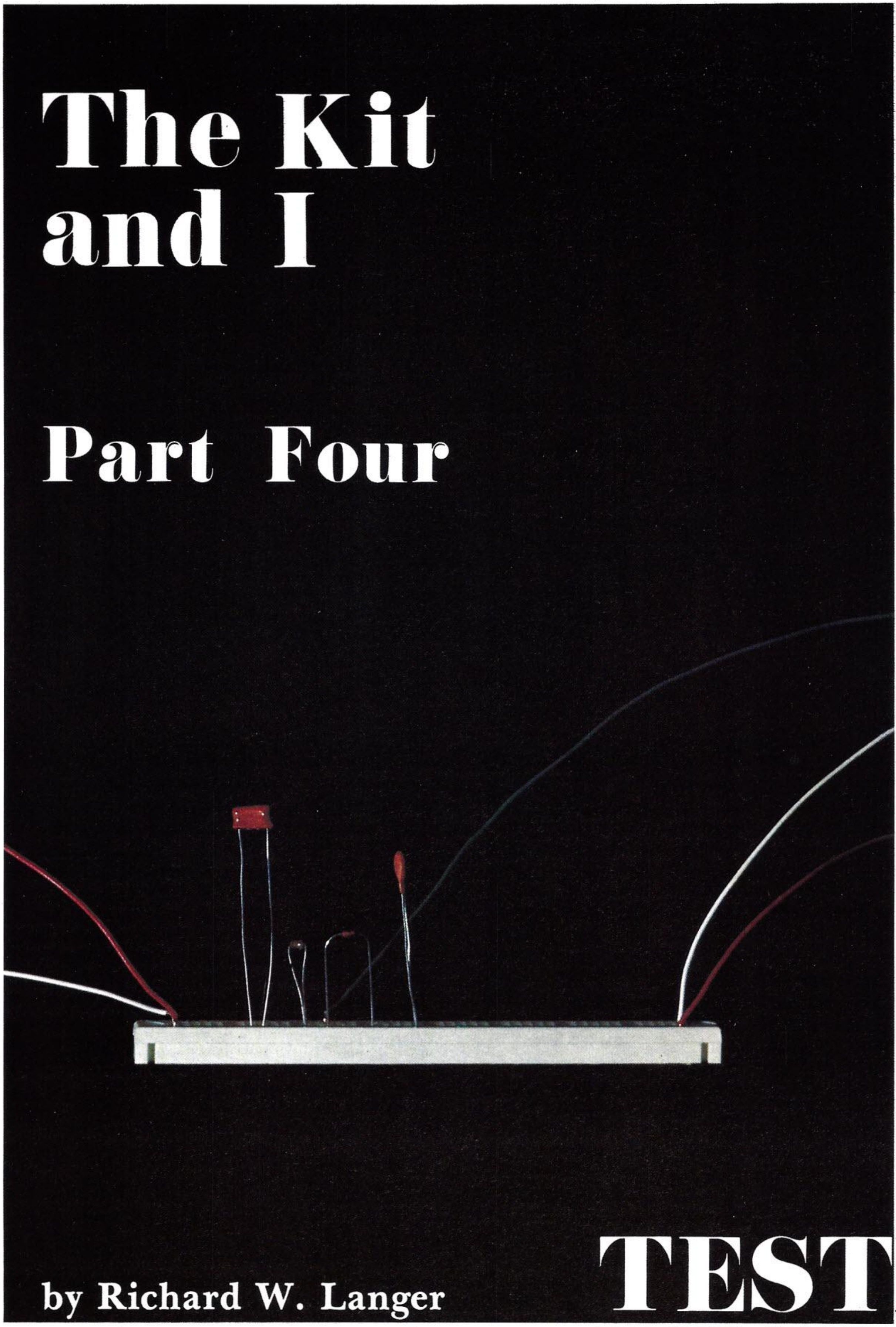
Processor Technology

The Kit and I

Part Four

TESTING TESTING

by Richard W. Langer



The results looked like something only an electrified Rube Goldberg could love.

Returning to the farm after shopping around for an oscilloscope, with many second thoughts (precipitated mostly by the price, which was about three times again what I thought it would be), I was greeted by another missive from Processor Technology. More corrections for the manual. As a timely note, they included a revised set of wave forms for the oscilloscope. That did it. If they couldn't make up their minds what the wave forms would look like, I certainly couldn't help them. So much for the oscilloscope.

The alternative was to construct a test probe according to Figure 3-3. "Assemble the probe using tack soldering technique," the instructions told me, leaving the rest up to my imagination. I'm sure "tack" has a totally different meaning in this context than our usual association with horses out here in the country. Whatever it is, I did not bother to find out . On my last visit to the local Radio Shack, I had seen these clever little boards for plugging components into on a temporary basis; I returned and purchased one.

Assembling the probe took only minutes, though the results looked like something only an electrified Rube Goldberg could love.

Before use, the probe had to be well grounded. The manual emphasized the point, though again without any specifics. I pondered the Sol-PC board wiring diagram for a while, looking around for a ground—considerably longer than I had to actually. The moment of truth was near. Sooner rather than later I had to face the fact of plugging in the whole affair. Sighing, I attached the ground.

Following a trip out to feed the fat chickens which had been fed before, I stepped bravely up to the workbench and plugged in the power supply. Holding my breath, I pushed in the start button, waiting for the whole PC board to go up in a shower of sparks. The fan started up slowly, reaching a steady hum in a few seconds. Suddenly something popped.

A tiny sliver of silver wire sailed through the air, landing next to U44. I hit the stop button, unplugged the unit, and made a quick visual inspection. Except for the sliver, nothing but an electronic smell. I had no idea from where the sliver popped nor whence the smell emanated. My nose was stopped up with a cold acquired while taking the bird nets off the blueberries in the rain the day before—three weeks late. So focusing on the origins of the odor was impossible.

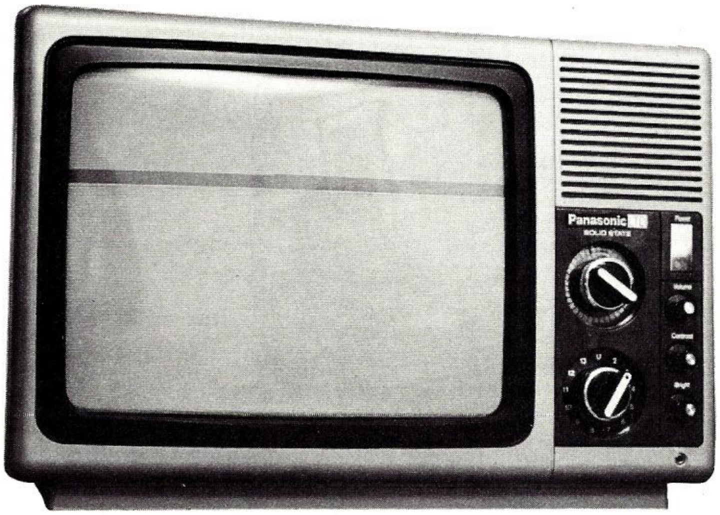
Plugging the power supply in again, I tried once more to run through the voltage test. The instructions indicated that "at pin 7 of U77 you should measure approximately 1.5 VDC." Fine and dandy. Even I knew it takes two to tango when it comes to electricity. If I put one of the probes on pin 7, where did I put the other?

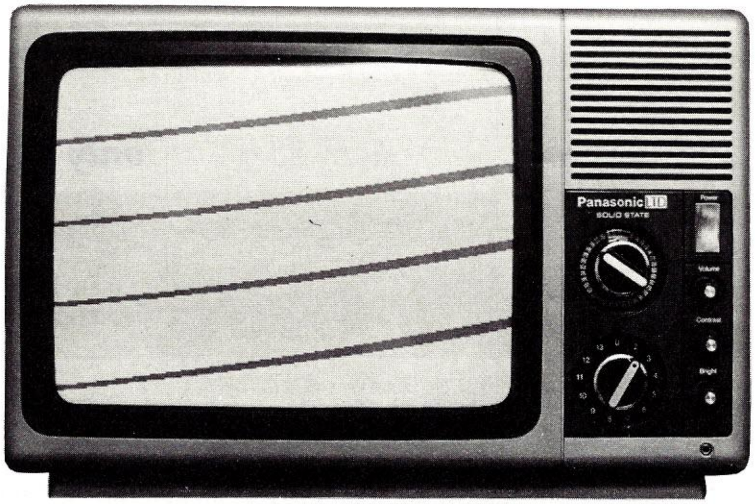
Studying the wiring diagram, I noticed that pin 1 was always indicated by a square while all the other pins were diagrammed by a round dot. Not much to go by. Still it was consistent, no matter how many pins the socket had, so pin number 1 was obviously unique.

Logic, if not necessarily reality, dictated that I put one probe on pin number 1 and one on number 7. Nothing happened. I reversed the probes. Zilch. Back to trying to conceive of the problem visually. I stared at the board. At least the power supply was all right. I knew that because I'd tested it before. If anything, it was a little high in one line—an error which I'd been told would correct itself once an extra bucking transformer arrived from the factory. (Not all the original transformers shipped, it seems, were up to design specifications.)

Well, if the power supply was all right, then the problem had to be on the board. On the other hand, I was making an assumption again. The power supply had been all right when I tested it. But that had been some time

Photographs by Thomas Hall





ago. The unit had been jostled around a bit in the meantime. Maybe something had come loose. A close inspection revealed nothing obvious. So I decided to test the power supply at the output, where it plugged into the PC board.

Now the plug was clearly designed so it could not be put in upside down, thus reversing the voltages. On the other hand, it was possible to attach the plug to the connection just to the left of where it should be—as, of course, I'd managed to do. Checking over my shoulder to see if anyone from Processor's warranty department had managed to sneak into my study (some three thousand miles from their home office), I unplugged the power supply and surreptitiously plugged it in properly. They'd never know.

Well, pin 7 of good old U77 showed a little life now. A little too much life, actually, since the designated 1.5 volts turned out to be more like 2.5. However, the warning specified only the danger of a "significantly lower" reading, so I let it pass. Proceeding down the test sequence for the other pins, everything worked out fine—if you disregarded the fact that all the voltages were higher than they should have been. Bingo, it was time to install another fifty or so resistors. R1, R2, R3, R4....

Something popped. A tiny sliver of silver wire sailed through the air, landing next to U44.

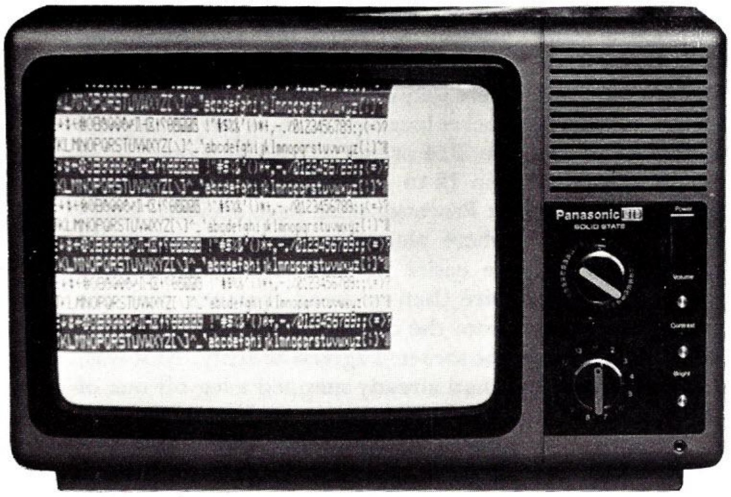
But why did the 1.5K resistors, brown-green-red bands, according to the manual, have a red dot between the red and gold band? What secret mark was it? What hidden meaning did it have? On the first one I thought it was just a sloppy paint job. However, I later noticed that all of them had manifestations of chicken pox, a fact which definitely implied intention, not accident. The meaning of the dot codes, however, still remains a mystery.

The positioning instructions for R80 had another one of those dandy little footnotes that accompanied D12. "The leads of R80 and its mounting holes form a snug fit. Take care when installing this resistor." I was all set to cobble up new leads by soldering on smaller wires as I'd been forced to do with D12. On the off chance that I might be able to ram R80 into place without the extra stilts, I gave it a try. The orange-orange-brown resistor fit about as snugly as a dime in a quarter slot. Looking around for something else along the orange-orange-brown line, I found one resistor larger than the rest. From my limited electronics knowledge, acquired since the beginning of this project, I assumed that the size was a product of its having been produced by a different supplier than the rest, particularly since there was no reference to any size difference in the manual. So there should be no electrical difference. And, since the silkscreened pattern on the board indicated a slightly larger resistor than the others, I chose to slip this mutant into R80. Somehow I was beginning to realize that precision electronics isn't always that precise.

Further down the resistor ladder, I came across R127, a 10K resistor coded orange-black-orange. Try as I might, I couldn't find it amongst the bag of parts Processor sent me. The same held true for R129. Wait a second, I thought to myself, 10K, that's supposed to be brown-black-orange. At least it always had been before. Since there were still plenty of these around, I assumed there was a color coding error in the manual and I used the brown-black-orange resistors.

At the end of this particular list of resistors to be coddled into place was a totally new animal. At locations VR1 and VR2 I was supposed to locate something with a 50K value. The color code was "potentiometer." Now I happen to have a thing about color—for years I conceived of chartreuse as a shade of blue, and mauve as rather greenish. You can't really build up true color identification, at least with the finer shades, until you make a visual association. For example, chartreuse instantly became a vivid shade of green as soon as I had my first glass of Chartreuse. (Just to confuse matters, the monks also turn out a yellow version.) But "potentiometer" was one color I'd definitely have to see in order to visualize.

By a process of elimination too involuted to explain (it lasted longer than the full playing-time of Shubert's Ninth, not counting time out for dashing over to the house to gather up another frosted toddy), I settled for a couple of strange black wheels with arrows on them to fill the spot. To start with, they were in a baglet full of parts unrelated to the ones I was working with. Second, they had three leads. Third.... Half a dozen extra pictures in the manual could make life so much easier.



With all the R2s and D2s to put into place I'd expected to meet him in person halfway through the kit. But all I came across now was another covey of strange capacitors—aluminum electrolytic, disk, monolithic, mylar tubular, none of them could stop me. I had this whole thing down to a science. What did stop me, however, was the fact that my wife doesn't use nail polish.

Checking over my shoulder to see if anyone from Processor's warranty department had managed to sneak into my study....

Seems I had to install the coaxial video cable. No problem. I had already put another chunk of the same cable onto the power supply, so I knew how to strip the outer insulation off the shield, dig out the inner conductor, and the like. But this time I ran into a huge caution note. After soldering the cable into place, I was supposed to cover the braided part with silicone glue (of which I had none) or nail polish (of which I also had none) to prevent fine bits of wire from working loose and "creating hard-to-find short circuits." I had enough problems with easy-to-find short circuits. Time for yet another trip to town.

The hardware store was closed. In fact the only thing open was a drugstore. Pulling in my stomach and pushing back my shoulders, I strode up to the cosmetic counter. "Excuse me. Could I please have a bottle of nail polish?"

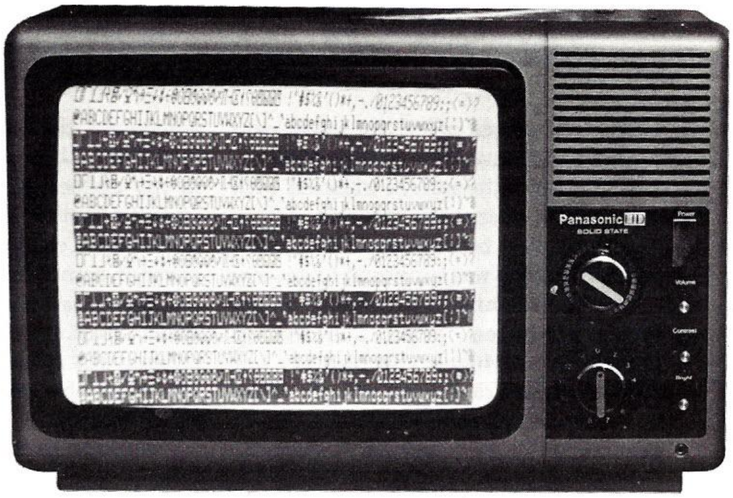
"Certainly." Was there a strange point to her smile? "What particular color were you looking for?"

"Oh, I don't know. How about something along the lines of capacitor red?"

"Pardon?"

"Never mind. I think she said clear."

Up till now I'd handled everything except the ICs themselves. But avoiding these sensitive devices, accompanied by so many notes of caution about static electricity, was no longer possible. Gingerly I slipped the specified ones into place. That is, I began gingerly. In no way were they



about to be pushed into their sockets gently. In fact, it wasn't unlike trying to slip boots over my kids' shoes. (Being last year's boots and this year's shoes, the fit is excruciatingly tight.) But, after a few tries, I learned how to rock them into place—the ICs that is—the boots I still haven't conquered completely.

Time for another volt-ohm meter check. Pin 12 of U28 was supposed to measure approximately one volt. Well, U28 was my Waterloo in the battle of the Sol. The voltage reading was barely one fourth of what it should have been. Now I was faced with that oft seen but previously unheeded phrase, "If you get a significantly lower reading, find and correct the cause before you proceed with assembly." But how? The manual gave not the slightest clue.

Turning the page, I discovered that but for that little malfunction of old U28 I could have finally seen some results on the video monitor. Granted it was only a test pattern. Still, I'd been no more than a hairsbreadth away from action when I tripped over pin 12 on U28. Would Processor know? I mean, how could they? I ignored Waterloo and continued.

Only one problem. They said to "connect Sol-PC video output cable to video monitor." All I had at the end of the cable was a sawed-off stump. Some kind of jack, which the instructions conveniently failed to mention, must be required. I scrounged around in my collection of parts-filled plastic bags and found some pieces that looked as if they'd cobble up into something appropriate. Skimming the manual, I also came up with the appropriate assembly instructions—some two sections and thirty pages after the cable was supposed to have been plugged in.

I was beginning to realize that precision electronics isn't always that precise.

Turning on the tube, I whistled with purposeful absent-mindedness for a minute or two before tapping Sol's on button. What I was supposed to get was a black horizontal bar moving slowly down the screen. What I ended up with was a black horizontal bar moving slowly down the screen. That is, it moved one third of the way down the screen. Then, kachonk, it stopped. No matter what I did, that's as far as it would go. On the other hand, the horizontal hold adjustment, which was supposed to produce an out-of-sync raster, did. Once I checked a dictionary to find out what a raster was, that is.

Still I could go no further. U28 was coming back to haunt me. I remembered it had tested out at only a third of the power requirement it should have. The horizontal bar would roll only one-third of the way down. There must be some connection. Dejectedly I called one of the engineers at Processor to confess my sins.

U28 was my Waterloo in the continuing battle of the Sol.

My *mea culpas* were for naught. Processor informed me that the bar's refusal to descend below the one-third level was an idiosyncrasy of the particular TV set they'd shipped me and nothing to worry about. Back to installing more ICs. Not having admitted the U28 fiasco to Processor, I could worry about it later.

Now the numbering of ICs is a secret art known only to the manufacturers. I'm not sure whether it's because some of them were made in Singapore, some in Taiwan, and some in Equador, or whether it's because engineers like to make life interesting, but none of the ICs were numbered in the same way. Sideways, upside-down, wrapped around, segmented, they seem to use any and all methods including a combination of cuneiform and hieroglyphics. Particularly hard to find were those numbered 74LS157. So difficult were they to locate, in fact, I became convinced that the manual actually meant 74LS175, a type which abounded in my collection. So convinced did I become that I substituted. Naturally, after I had completed the task, I discovered some ICs with 157 scribbled on their backs in barely noticeable silver lettering. I might add that removing ICs from their sockets is more of a pain than placing them there in the first place.

I mention that only because the next instruction, after I'd managed to put the proper ICs in their places, began, "Remove U42 and bend pin 6 out forty-five degrees to its normal position." Obviously they meant "from" its normal position. But who was I to argue? I did what they meant, not what they said. The question remains, why didn't they have me bend pin 6 when I first slipped U42 into place? It could have saved some fingernails.

It wasn't enough that I had to pry U42 loose and contort its legs. Oh no, U59 had to have its fourth leg lifted, and U75 its fifth leg. The whole board was beginning to look like an electronic zoo with numerous chips running around looking for a convenient fire hydrant.

A couple of jumpers were required to close some circuits on as yet unoccupied sockets so that a temporary test pattern could be made to appear on the screen. The need for jumpers increased the chaos of the whole affair, but at least it saved me the task of putting yet more ICs into place and then prying them loose again. There were only two points to connect at each socket location, such as pin 12 to pin 6 on U14 or pin 12 to pin 5 on U15 or pin 12 to pin 4 on U16, and someone at Processor must have decided, somewhere along the line, that it would be easier to cross the points with a wire than to bend fourteen pins out from the chips, leaving two to fit into the socket. I agreed heartily. As it was, I had already snapped a leg off one of the 74LS175s by rocking it in and prying it out of the wrong socket. I followed this act up with a minor triumph of first-timers-don't-really-know-what-they're-doing-but-why-not-try-it homebrew engineering: I studied the wiring diagram until I located a 74LS175 position where pin 8, the one I broke off, appeared to connect to nothing.

The only problem with the jumper step was that I had none of the #24 wire specified for the task. I wasn't about to go into town again that week. So I did what I always do on such occasions. I made a pilgrimage to the workshop next to the garage. Therein is vaulted a true treasure.

The needle didn't exactly jump when I ran the first test on pin 7 of U77. In fact, it wouldn't budge. The circuit was dead.

When Susan's parents moved to smaller quarters, I inherited the "junk" from their garage—an incredibly useful collection of spare parts and tools collected by my father-in-law over the course of half a century. There's Okonite Tape in the original 1930s tin. There's a complete set of rusted wood chisels ensconced in an old cedar cigar box emblazoned with the University Club of Chicago seal. There's a sliding-top one-pound box of Old English Choice Salt Cod Fish now filled with fuses that appear to have been constructed by Edison himself. And there's wire. Brass wire, copper wire, lead, steel, and aluminum wire, as well as several others of indefinable composition. But nothing labeled #24. And if the truth will out, I wasn't sure if the "#" represented "pounds" or "number."

Chucking the whole idea, I headed back to my own much younger but rapidly burgeoning collection of junk at the study and extracted a slew of clipped-off component leads. If they were good enough for the components, certainly their quality for use as temporary jumpers could not be faulted. So into the appropriate sockets they went.

Now came the real moment of truth. No more calling up a black band on the video monitor. I was supposed to get a whole alphanumeric test pattern. Did I dare hope?

At last I flicked the switch. There it was, just like in the picture—except someone had punched in a side and it swooped. For no particular reason I thought of turning the little arrows on the potentiometer. Everything came up perfect.

I stared and stared at the test pattern, not believing I'd really done it. Slowly, however, my elation gave way to disappointment. There it was, right in front of my nose. All those numbers and letters on the video screen. They'd come from the Sol board I'd assembled. But how? In which one of those black chips did they keep the letters? And how did they get them from there to the screen? Even having put most of the computer together, I still didn't really understand. ▼

PUTTING TWO & TWO

PART 0010

by Tom Pittman

In the fourth issue of *ROM* (October 1977), we looked at binary numbers from a mathematical point of view, and at binary arithmetic for personal computers. We saw how addition and subtraction work; we looked at Two's Complement notation for negative numbers, and we observed integer multiplication and division. These are fundamental concepts for the correct understanding of all computer arithmetic. If you feel a little hazy on the subject you may wish to go back and review that article before continuing. On the other hand, if you have a good grasp of binary addition and are looking only for a general understanding of decimal and floating-point arithmetic, read on.

When we looked at binary arithmetic, we noticed that the same rules and procedures which we learned in grammar school could apply, except that the radix is two, not ten. As it turns out, most of the microprocessors available to us today can also add in decimal. Or so it seems.

First let me remind you that the computers made today are all binary machines. What do I mean by "add in decimal?" We need some way to represent decimal numbers in binary. We call this "Binary Coded Decimal" or BCD; each of the ten decimal digits is represented by four binary digits as follows:

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

You may notice that there are six other possible combinations of four bits which we do not use. They are called *undigits*. You will also notice that the decimal value assigned to each four-bit number corresponds to its normal (counting) value in binary. This makes the arithmetic easier to understand, though not necessarily easier to build hardware for. Another popular representation is called "excess-3" or XS3, and uses a different set of representations which makes the hardware simpler. This is not used in microprocessors, so I will not say any more about it.

What happens when we want to show more than one decimal digit? We simply put the corresponding BCD digits together:

47 = 0100 0111
82905 = 1000 0010 1001 0000 0101

Illustrated by Linda Smythe

TOGETHER

For eight-bit microprocessors it is usually convenient to pack two digits into each byte, so the following discussion will assume this format.

Suppose we want to add 47 + 21. Remembering that the computer does its addition in binary, let's see what happens:

```
  47      01000111
 +21     +00100001
 ---     ---------
  68      01101000
```

Gee, that was easier than we thought. Maybe decimal arithmetic is not so hard after all. Try 47 + 39:

```
  47      01000111
 +39     +00111001
 ---     ---------
  86      10000000  (=80)
```

As you can see, the correct answer is 86, but the calculated value (in binary) is 80. Clearly, unadulterated binary arithmetic will not do the job. Looking at this problem more closely, we notice that whenever there is a carry out of the right-hand digit, the result is low by a value of six:

```
  12      0001 0010
 +34     +0011 0100
 ---     ----------
  46      0100 0110  (no carry, sum OK)
```

```
 (1)       (1)
  18      0001 1000
 +19     +0001 1001
 ---     ----------
  37      0011 0001  (carry, sum low)
```

In fact, even if there should have been a carry (in the decimal addition) but was not in the binary imitation, we come out low:

```
 (1)       (0)
  37      0011 0111
 +45     +0100 0101
 ---     ----------
  82      0111 1100
```

In this case, however, the sum is not even a valid decimal number. Thus we can begin to formulate a rule for fixing the result of trying to add BCD numbers in binary hardware. For each digit of the sum, if there is a carry out of that digit, or if the sum is an undigit, add six. Any carries out of that digit, of course, have to be added into subsequent digits (to the left). Let's look at those sums again:

```
  37      0011 0111
 +45     +0100 0101
 ---     ----------
          0111 1100  (undigit, add 6)
         +0000 0110
         ----------
  82      1000 0010
```

```
  18      0001 1000
 +19     +0001 1001
 ---     ----------
          0011 0001  (carry, add 6)
         +0000 0110
         ----------
  37      0011 0111
```

Now we can also see it work for the left-hand digit:

```
  55        0101 0101
 +76       +0111 0110
 ---       ----------
            1100 1011  (undigit, add 06)
           +0000 0110
           ----------
            1101 0001  (undigit, add 60)
           +0110 0000
           ----------
 131      1 0011 0001
```

```
  96        1001 0110
 +64       +0110 0100
 ---       ----------
            1111 1010  (undigit, add 06)
           +0000 0110
           ----------
          1 0000 0000  (carry, add 60)
           +0110 0000
           ----------
 160      1 0110 0000
```

To help you know when to add 06 or 60, the microprocessor has a special memory to save the carry out of each of the two digits. The carry out of the left-hand digit is the same carry flag we have come to know and love in binary arithmetic. The right-hand carry is called a *half-carry* (because it occurs at the half-byte boundary), and has its own position in the *status word,* the computer word that contains information used to control processing. A special machine instruction is provided in these CPUs (Central Processing Units) to see whether either digit is an undigit, to look at the two carry flags, and then to add the appropriate combination of 06 and 60. This instruction is usually called *decimal adjust.* It does its trick in one fell swoop, rather than in two steps as we did above. So to do BCD addition in such computers you do a binary addition, followed by the decimal adjust, and the result is correct.

Some microprocessors do not have a decimal adjust instruction; instead they have direct decimal arithmetic. In such cases you can think of the decimal adjust as being built into the ADD instruction. Note that most of the arithmetic is still done in binary. A few microprocessors don't do decimal arithmetic at all, lacking the necessary hardware to do so. It is important to realize that without a hardware half-carry sensor it is impossible to do Packed BCD (two digits in one byte) decimal arithmetic. You can still write programs to add one digit at a time, simulating the hardware as we did on paper.

Adding in BCD is, as we have seen, made fairly easy by the hardware manufacturers. Subtracting is another story. Some make it easy—you do the same kinds of steps as in adding. But more likely the procedure is circuitous.

Remember, subtracting is adding the negative. If Two's Complement gives us the negative of a binary number, then Ten's Complement should give us the negative of a decimal number. The Two's Complement negative of a number n is $(256-n)$, or rather, $(255-n+1)$. The Ten's Complement of n is, similarly, $(99-n+1)$. Clearly, subtracting any decimal number from 99 introduces no carries or borrows, and the result is always in true decimal (no undigits), so there are no adjustments to make. We add one at the same time that the negative is added to the minuend. Let's look at some negatives:

$$\begin{aligned}
-47 &= 99-47+1 = 52+1 = 53\\
-99 &= 99-99+1 = 00+1 = 01\\
-00 &= 99-00+1 = 99+1 = 100 = 00
\end{aligned}$$

In the case of -0 you see the reason for saving the $+1$ part for the sum. Actually we can cheat a little, since the computer is really doing its arithmetic in binary, and does not mind a few undigits in embarassing places. Add the $99+1$ first:

$$99+1 = 10011001 + 00000001 = 10011010$$

I cannot show you what this looks like in decimal, because it is not legal BCD, but,

$$\begin{aligned}
-47 &= 10011010 - 01000111 = 01010011 = 53\\
-99 &= 10011010 - 10011001 = 00000001 = 01\\
-00 &= 10011010 - 00000000 = 10011010 = \,.\,.
\end{aligned}$$

Now, to subtract we need only add, then decimal adjust the result:

$$\begin{aligned}
82-47 &= 82+53 = 135, \text{ or 35 with a carry (positive)}\\
16-99 &= 16+01 = 17 \text{ (no carry, so negative)}\\
&= 17-(100) = -(99-17+1) = -83\\
24-00 &= 24+\,.. \text{ (hmmm, better try this in binary...)}\\
24-00 &= 00100100 + 10011010\\
&= 10111110 + \text{dec.adj.}\ (66)\\
&= (1)00100100 = 24 + \text{carry (positive)}
\end{aligned}$$

I know. It looks like a *kludge* (an awkward or "inelegant" way to do part of a program). But it is mathematically correct, and you don't have to do it very often.

As with binary numbers, it is frequently the case that one byte will not contain a large enough BCD number to be interesting. Fortunately, everything I said about multiple precision binary numbers applies also to BCD. Since you are actually using the binary ADD instruction of the computer to do the decimal arithmetic, you can as easily use the add-with-carry instruction to process additional digit pairs. Those nice people who designed the microprocessors were careful that the decimal adjust leaves the normal carry flag in exactly the correct condition so that it can be added into the next byte; in such a case it truly is a decimal carry (meaning, number greater than two digits). For multiple precision BCD subtraction, you have to be a little more careful not to confuse the carry and borrow significances of the same flag. Since the subtraction is done with an ADD instruction, not a SUBTRACT instruction, the carry flag is a true carry (that is, as we observed in the previous article, the complement of a borrow). When you do the next byte, add it to 99, then subtract the next byte of the subtrahend:

$$\begin{array}{rr}
5234 & 01010010\ 00110100\\
\underline{-1609} & \underline{-00010110\ 00001001}
\end{array}$$

First, Ten's Complement right byte and add:

$$\begin{array}{lr}
 & +10010001\\
 & +01010010\ 00110100\\
\hline
 & 11000101\\
\text{(decimal adjust)} & +01100000\\
\hline
 & (1)00100101
\end{array}$$

Now compute Nine's Complement of next byte plus carry:

$$\begin{array}{lr}
 & 10011001\\
 & +1\\
 & -00010110\\
\hline
 & 10000100\\
\text{(add minuend)} & +01010010\\
\hline
 & 11010110\\
\text{(decimal adjust)} & +01100000\\
\hline
3625 & (1)00110110\ 00100101
\end{array}$$

The carry out here reminds us that the result is positive.

I am not going to spend much time discussing decimal multiplication and division. I think you can see what needs to be done. Remember, the shift-and-add or shift-and-trial-subtract techniques require that you shift a whole digit position (four bits for BCD, not one) and that the multiplication table is somewhat larger. Usually the decimal multiply routines in computer programs use a decimal add subroutine, calling on it repeatedly while counting down on a single digit of the multiplier. Then the partial product and the multiplier are shifted one digit position, and the process repeated. The divide reverses the process, subtracting the divisor decimally from the left end of the dividend while counting up on the left digit of the quotient. Because a shift of one BCD digit (four bits) is awkward on most microprocessors, you will not see much of this. This, incidentally, is why the BASICs with decimal arithmetic are so much slower than their counterparts with binary arithmetic.

If decimal arithmetic is so bloomin' hard, why bother at all? One reason is that people think in decimal, not binary. However much more efficient binary may be than decimal, computers should serve people rather than people serving computers. But all is not lost. In most cases (we will see the exceptions shortly) it is perfectly adequate for the computer to do its calculations in binary, and to convert its input from decimal to binary and its output from binary to decimal. In most such cases the human beings using the system need never know the difference.

As usual when discussing some mathematical task for the computer, let's first see how we (as human beings) would do the same thing. If you are lazy like me, you have a little chart which makes a correspondence between decimal numbers and binary. To convert from one to the other you look up the corresponding number in the table. Charts, unfortunately, waste computer memory.

Since decimal and binary are both ways of representing counting numbers, we could count up on one scale while counting down on the other. This is something like doing arithmetic by counting on your fingers. Don't laugh; it is a valid, though slow, way of doing it.

Now let's be serious. There has to be a better way to do it, and there is. The trick is to convert one digit at a time. Weighting factors are applied to each digit of a number (as explained in the October issue of *ROM*). If the number is decimal, the weights are (from right to left) 1, 10, 100, etc.; if the number is binary, the weights are (in the same order) 1, 2, 4, 8, etc. So for each digit we pick off of one number, we multiply it times its weight and add it to the other number. Take the binary number 10011100110. The weights are:

1024 x 1	1024
512 x 0	
256 x 0	
128 x 1	128
64 x 1	64
32 x 1	32
16 x 0	
8 x 0	
4 x 1	4
2 x 1	2
1 x 0	
	1234

But if you try to write a program to do it exactly that way, it will be a monstrous kludge. Consider instead how you would do it with a calculator (if you are lazy like me, and want to save steps). Start with zero on the calculator display, and the binary number written on a piece of paper. Repeat the following two steps until the binary number is used up:

1. Multiply times 2 on the calculator.

2. Add the leftmost digit of the binary number, and cross it off.

Now look at what you did. Notice that the first bit was added as a 1. But there were ten multiply-by-two steps after it (because there were ten more digits), effectively multiplying that 1 times 1024 (2 raised to the tenth power). The next two bits were added as zeros, which is the same as not adding. The fourth bit was followed by only seven multiplies, and so on. Try it. It is a bit tedious, but it works. And the computer likes tedious work.

Unfortunately, we went through this little example using our normal thinking arithmetic, which is decimal, not binary. So is the calculator, but it does not help in programming a binary computer. However, all is not lost. Remembering (as we noticed so often in the October article) that if we can isolate the procedures from the values by changing the values from decimal to binary, we can use the same procedures in the computer. Ummm, also change the values from binary to decimal: it is a complete dualism. That means that we will be converting decimal numbers to binary in the computer. For example, take the number 1234:

1111101000 x 1	1111101000
1100100 x 10	11001000
1010 x 11	11110
1 x 100	100
	10011100110

Remember, that was the hard way. Now the easy way—write down the decimal number on the computer equivalent of a piece of paper. We might call this a memory register, but its name is not important. Start with zero in the register(s) where the answer will be. Then repeat the following two steps as long as there are decimal digits left:

1. Multiply the result register by ten. You can do this by shifting it left two bit positions, adding the original value, then shifting left again.

2. Add the leftmost digit of the decimal number to the binary result, and cross it off. You can do this by shifting the number over by one digit position, or by advancing the digit pointer.

When you finish, the result register contains the binary equivalent of the decimal number with which you started.

"Fine," you say, "but how do I get back?" I thought you'd never ask! Simply reverse the process. Carefully observe what we did. We shifted the starting number left, eating the digits. So in reverse, we will shift the ending number right, spitting out digits. The next step was to multiply one number by the radix of the other number. In reverse we divide (the remainder is the next digit). Got that? To shift from binary to decimal in a binary computer, you divide by ten, and the remainder is the next digit (starting from the right). Then divide the quotient by ten again, and so on.

If you are working with sixteen bits or less, sometimes it's easier to count the number of times you can subtract 10000 from the number, then count the number of times you can subtract 1000, then 100, then 10; what's left is the fifth digit. I have also seen tricky algorithms—an algorithm is a set of well-defined rules used for solving a problem in a finite number of steps—which divide by multiplying (I never really quite understood it, but the idea was to multiply times one tenth).

My favorite conversion routine uses the decimal add feature of the microprocessor, and comes out clean and fast. Remember when I suggested doing it with a calculator? Well, that is exactly how it is done, only by the computer. Picking off bits is as easy as shifting left. Multiplying by two (in decimal, mind you) is adding the number to itself using the decimal adjust. You use the add-with-carry for the whole number, because the carry into this multiply is just exactly the carry link bit that was shifted off the left end of the binary number.

I did not mention signed numbers in the discussion of number radix conversion. I think you will find the signs to be a simple matter of remembering the sign, converting the (positive) value of the number, then putting the sign back on.

So far all the numbers we have discussed are integers. That is fine up to a point. But soon we discover that the real world is not all 3-4-5; many things are fractional. Consequently, our computer programs, which often must reflect the real world, need fractions too. An obvious example is the use of the computer in the computation of sums of money. Before we go any further, I want to stress the fact that the computer arithmetic built into the hardware deals only with integers—whole numbers. How, then, do we handle fractions? In exactly the same way as any other concept not specifically built into the computer: we simulate the fractions. Instead of computing (as we would like) whole and fractional dollars, we compute whole cents. The number 53.84 is represented in the computer by the integer 5384, or rather by the binary integer 1010100001000. That is, we compute on 5,384 whole objects, which themselves happen to be hundredths of the real objects we are interested in. We can in fact do this for any rational fractions: 5.6 is 56 tenths; 2¾ inches equals 11 quarter-inches; an hour and twenty minutes is 80 sixtieths of an hour.

As you will recall from grammar school arithmetic, when you add fractions, you have to put them into what we called "improper form" consisting of one integer divided by another. We have already done that when representing them in the computer. Specifically, we remember the denominator (in our heads) and compute on the numerator. The second step in adding fractions is to be sure they all have the same denominator: 3/4 + 1/2 is converted into 3/4 + 2/4. Now, as you can see, all of our numbers are still integers. So much for fractions.

Alas, real numbers are not always nice, not even fractions. In the fourth grade they let us use 22/7 for pi, but in the sixth grade they told us that it is not exact (it is wrong in the third decimal place), and by now you know, of course, that pi cannot be exactly represented by a fraction at all. What do we do? We fake it. We take several bits (between 20 and 100), and we say that is close enough. I'll tell you how.

You know how to handle particular fractions in the computer. We are going to process all real numbers in the same general way. The computer will treat each number as an improper fraction, consisting of a numerator and a denominator. The numerator will always be some standard number of bits, say twenty-four (three bytes are very common). Because we do not want to waste memory space, the denominator will be in a special form. We pick some number that is convenient, like two or ten or sixteen, and say that the denominator is always some power of that number. This way we need only remember the power of that number, giving us a very large range of possible denominators for a relatively small number of bits (usually seven or eight). The number we are choosing the powers of is nothing more, or less, than the radix of our arithmetic. When the radix is ten, we are doing decimal arithmetic, and the form we have chosen for the representation of real numbers is what is often called *scientific notation*. For example, if you look up the mass of a proton, the chemical handbook will say,

$$1.67 \times 10^{-24}\,\mathrm{g}$$

This means that the numerator is 167 and the denominator is 10 raised to the power 24. Notice I moved the decimal point, so there are no real fractions left. Look up the speed of light; the book says:

$$2.9979 \times 10^{8}\,\mathrm{m/sec}$$

The numerator is 29979, and the denominator is 1/1,000,000. As you see, I fudged a bit—we have a fractional denominator. Actually, that is not a problem because we are actually storing in the computer not the value of that fraction, but rather what that fraction tells us about where the decimal point belongs in what I called the numerator.

There are special buzzwords to describe the two parts of this number—the numerator is generally called the "fraction" and the denominator is generally called the "exponent" part. As these words imply, the number is stored in a slightly different form than I have described. Instead of treating the numerator as a true integer, we pretend that it is a decimal fraction, with the decimal point just to the left of the left-most significant digit. The number of places that the decimal point needs to be moved to the right for the number to be correct is the same as the exponent on a power of ten. Note that if the decimal point needs to be moved to the left, the exponent part will be negative. I realize that this sounds different from the first description I gave you, but conceptually it is the same. If we are talking about a six digit fraction part, the computer (for all it knows) is only dealing in integer millionths; the position of the decimal point is nothing more than a figment of our imagination. And whether we consider the exponent part to be a multiplier or a divisor is again only a question of whether the sign is positive or negative. However, because we insist that the decimal point be assumed on the left of the number, the number representation is often called *floating point,* especially since the decimal point seems to "float" on top of, or to the left of the fraction.

Doing arithmetic in floating-point numbers is exactly the same as doing arithmetic with fractions. In order to add or subtract, line up the (actual) decimal points, then add or subtract as integers, for example:

$$.1234 \times 10^{56} + .3412 \times 10^{53}$$

$$\begin{array}{ll} .1234 & (56) \\ \underline{+.0003412} & (56) \\ .1237412 \times 10^{56} & \end{array}$$

$$.7654 \times 10^{-18} - .7648 \times 10^{-18}$$

$$\begin{array}{ll} .7654 & (-18) \\ \underline{-.7648} & (-18) \\ .0006 & (-18) \end{array}$$

which is

$$.6000 \times 10^{-21}$$

The last step in the second example is what we call "normalizing" or putting the number back into its normal form, with the decimal point floating to the left of the fraction.

Multiplying is a little easier. Looking at it algebraically, we see that

$$(A x B) \text{ x } (C x D) = A x B x C x D = A x C \text{ x } B x D$$

Also,

$$A^{B} \text{ x } A^{C} = A^{(B+C)}$$

So all we need to do is multiply the fractions and add the exponents:

$$\begin{aligned} &.1024 \text{ x } 10^{23} \text{ x } .6250 \text{ x } 10^{-14} \\ &= .6400 \text{ x } 10^{8} \end{aligned}$$

Careful! I sneaked a normalizing step in on you. If you work it out you will see that the product of the fractions is actually .0640000, and that we normalized, then trimmed the answer down to four digits. If any significant digits had been out there, they would have been discarded. For this reason floating-point arithmetic is and remains only an approximation of the real numbers we pretend to deal with.

Dividing is, of course, quite similar to multiplying. You divide the fractions, and subtract the exponents. Oh, didn't I tell you how to divide fractions? It's the same as dividing integers, but instead of stopping when we have used up the dividend, we continue tacking zeros onto the end until the quotient is normalized. However, we need to keep track of how many extra steps we go through, so that the exponent of the quotient can be adjusted accordingly. Let's see how this works:

$$.2048 \text{ x } 10^{58} \text{ / } .8192 \text{ x } 10^{85}$$

$$\begin{array}{r@{}r} & \underline{0000.2500} \\ 8192.\,\big) & 2048.0000 \\ & \underline{1638.4} \\ & 409.60 \\ & \underline{409.60} \\ & .0000 \end{array}$$

Notice that if we had stopped at the end of an ordinary integer division, the quotient would be zero and the remainder 2048. Instead, we continued adding zeros to the end of the dividend, repeating the divide step until we had gone three or more digits (some of them zero) past the first nonzero quotient. When we do this on paper we have all the decimal points lined up to remind us where they belong; in the computer we have to count the divide steps so we don't lose track of the decimal point.

Thus far we more or less understand decimal floating point. This is great for decimal arithmetic in the computer, but as we noticed earlier, decimal is a lot more tedious and slower than binary. All that analysis was for nothing. Or was it? Remember again that the procedures carry across, but the constants are converted. Binary floating point is done with a binary fraction and a binary exponent. The exponent tells how many places to the right the binary point must be moved; it is the exponent of a power of two. Everything else is the same.

I suspect that the notion of a "binary point" may throw you, so let's think about it for a while. I assume you have no trouble correlating binary and decimal integers. Binary 110 is decimal 6. But what exactly does .6 mean in decimal? It is simply 6/10 because 10 is the radix. So, .1 in binary is 1/2 because 2 is the radix. By the same reasoning we can see that .03 is 3/100 in decimal, and .01 is 1/4 in binary. In the same way that the digits to the left of the decimal point have weights in positive powers of ten, the digits to the right of the decimal point have weights in negative powers of ten. So, too, the digits to the left of the binary point have weights in positive powers of two and the digits to the right of the binary point have weights in negative powers of two. What is the value of 110.101 in binary? It is 6 5/8. Now just as there are numbers which are repeating decimals, so there are numbers which are repeating binary fractions. Take 1/3. In decimal this is .33333.... In binary it is .01010101.... If you multiply 3 x .33333..., you will get .99999..., which is equivalent to 1. If you multiply 11 x .010101..., you will get .111111..., which is also equivalent to 1 (in binary).

Obviously we cannot store infinite decimals—I mean, binaries—in the computer, so we run up against the fact that .99999999 does not equal one. We call this problem *truncation error* or *round-off error* depending on what we did to the last digit. And if you get your friendly computer (preferably using BASIC) and tell it to divide one by three, and divide the quotient by three, and so on for a hundred divisions, then multiply by three hundred times, I can guarantee you will not end up with one. It probably won't even be close. Now do the same thing, but divide by ten. If your BASIC uses decimal arithmetic, the answer will be exactly correct; if it uses binary, it will be wrong. Next try it with two. The point is, that because we humans think in decimal, if our computers do their arithmetic (or at least the fractions) in decimal, we will get exactly the answers we expect. If, on the other hand, the computers compute in binary, the results will come a lot faster, and in most cases the answers will be every bit as accurate (no pun intended). However, we will not necessarily be able to add \$.35 + \$.65 and get exactly a dollar.

Obviously, adding, subtracting, multiplying, and dividing are not the only operations you will want your computer to do. There are square roots, trigonometric functions, exponentials, and logarithms, and you-name-it. But it is not the place of an introductory tutorial on computer arithmetic to cover them. If you really want to work with them, take a course in numerical analysis at your local community college. The second volume of Knuth's *The Art of Computer Programming* discusses some of these methods, but not in depth. I myself am still looking for some good reference works in this area. Perhaps it is because so many people have taken advice such as I gave at the beginning of this article that we do not have much published information on the subject. Still I think that while it is nice to know how these things work, if we all spend our time on basic mathematical routines, we will never get down to the business of making computers useful. So once you understand the basics, which is always good and curiosity satisfying, let the computer do the adding. There are better things for us to think about. And one of the benefits of computers is to take over the tedium of a complex technological society so we have time to think. ▼

COMPUTER MODELS IN PSYCHOLOGY

by Joseph Weizenbaum

Sometimes a very complex idea enters the public consciousness in a form so highly simplified that it is little more than a caricature of the original; yet this mere sketch of the original idea may nevertheless change the popular conception of reality dramatically. For example, consider Einstein's theory of relativity. Just how and why this highly abstract mathematical theory attracted the attention of the general public at all, let alone why it became for a time virtually a public mania and its author a pop-culture hero, will probably never be understood. But the same public which clung to the myth that only five people in the world could understand the theory, and which thus acknowledged its awe of it, also saw the theory as providing a new basis for cultural pluralism; after all, science had now established that everything is relative. A more recent example may be found in the popular reception of the work of F. Crick and J.D. Watson, who shared the Nobel prize in Medicine in 1962 for their studies of the molecular structure of DNA, the nucleic acid within the living cell that transmits the hereditary pattern. Here again highly technical results, reported in a language not at all comprehensible to the layman, were grossly oversimplified and overgeneralized in the public mind into the now-popular impression that it is already possible to design a human being to specifications decided on in advance. In one fell swoop, the general public created for itself a vision of a positive eugenics based not on such primitive and (I hope) abhorrent techniques as the killing and sterilization of "defectives," but on the creation of supermen by technological means. What these two examples have in common is that both have introduced new metaphors into the common wisdom.

Einstein's theory provided a new basis for cultural pluralism; after all, science had now established that everything is relative.

A metaphor is, in the words of I. A. Richards, "fundamentally a borrowing between and intercourse of thoughts, a transaction between contexts." Often the heuristic value of a metaphor is not that it expresses a new idea, which it may or may not do, but that it encourages the transfer of insights, derived from one of its contexts, into its other context. Its function thus closely resembles that of a model. A

It was easy for the public to see the "cracking" of the genetic code as an unraveling of a computer program.

From *Computer Power and Human Reason,* Copyright © 1976 by W. H. Freeman and Co.

Illustrated by Cindy Hain

Western student of Asian societies may, for example, not learn anything directly from the metaphoric observation that the overseas Chinese are the Jews of Asia. But it may never have occurred to him that the position of Jews in the Western world, e.g., as entrepreneurs, intellectuals, and targets of persecution, may serve as a model that can provoke insights and questions relevant for understanding the social role and function of, say, the Chinese in Indonesia. Although calling that possibility to his attention may not give the Western student a new idea, it may enable him to derive new ideas from the interchange of two contexts, neither of which are themselves new to him, but which he had never before connected.

Neither the idea of one object moving relative to another, nor that of man being fundamentally a physical object, was new to the common wisdom in the 1920s and the 1960s, respectively. What struck the popular imagination when, for some reason, the press campaigned for Einstein's theory, was that science appeared to have pronounced relativity to be a fundamental and universal fact. Hence the slogan "everything is relative" was converted into a legitimate contextual framework which could, potentially at least, be coupled to every other universe of discourse, e.g., as an explanatory model legitimating cultural pluralism.

The results announced by Crick and Watson fell on a soil already prepared by the public's vague understanding of computers, computer circuitry, and information theory (with its emphasis on codes and coding), and, of course, by its somewhat more accurate understanding of Mendelian genetics, inheritance of traits, and so on. Hence it was easy for the public to see the "cracking" of the genetic code as an unraveling of a computer program, and the discovery of the double-helix structure of the DNA molecule as an explication of a computer's basic wiring diagram. The coupling of such a conceptual framework to one that sees man as a physical object virtually compels the conclusion that man may be designed and engineered to specification.

What Minsky means by "understanding" music or painting is quite different from what Mozart or Picasso meant by it.

There is no point in complaining that Einstein never intended his theory to serve as one half of the metaphor just described. It is, after all, necessary for the two contexts coupled by a metaphor to be initially disjoint, just as (as I insisted earlier) a model must not have a causal connection to what it models. The trouble with the two metaphoric usages we have cited is that, in both, the metaphors are overextended. Einstein meant to say that there is no fixed, absolute spacetime frame within which physical events play out their destinies. Hence every description of a physical event (and, in that sense, of anything) must be relative to some specified spacetime frame. To jump from that to "everything is relative" is to play too much with words. Einstein's contribution was to demonstrate that, contrary to what had until then been believed, motion is not absolute. When one deduces from Einstein's theory that, say, wealth and poverty are relative, in that it is not the absolute magnitudes of the incomes of the rich and poor that matter, but the ratios of one to the other, one has illicitly elevated a metaphor to the status of a scientific deduction.

Psychologists seem to have misunderstood just what it is that makes physics somehow more a science than psychology.

The example from molecular biology illustrates an overextension of a metaphor in another sense; there the extent of what we know about the human as a biological organism is vastly exaggerated. The result is, to say the least, a premature closure of ideas. The metaphor, in other words, suggests the belief that everything that needs to be known is known.

The computer has become a source of truly powerful and often useful metaphors. Curiously, just as with the examples already cited, the public embrace of the computer metaphor rests on only the vaguest understanding of a difficult and complex scientific concept (here, the theory of computability and the results of Turing and Church concerning the universality of certain computing schemes). The public vaguely understands—but is nonetheless firmly convinced—that any effective procedure can, in principle, be carried out by a computer. Since man, nature, and even society carry out procedures that are surely "effective" in one way or another, it follows that a computer can at least imitate man, nature, and society in all their procedural aspects. Hence everything (that word again!) is at least potentially understandable in terms of computer models and metaphors. Indeed, on the basis of this unwarranted generalization of the words "effective" and "procedure," the word "understanding" is also redefined. To those fully in the grip of the computer metaphor, to understand *X* is to be able to write a computer program that realizes *X*. This is vividly exemplified by Professor Marvin Minsky, Director of M.I.T.'s Artifical Intelligence Laboratory, who writes,

> [For computers] "to write really good music or draw highly meaningful pictures will of course require better *semantic* models in these *areas*. That these are not available is not so much a reflection on the state of heuristic [computer] programs as on the traditionally disgraceful state of analytic criticism in the arts—a cultural consequence of the fact that most esthetic analysts wax indignant when it is suggested that it might be possible to understand what they are trying to understand."

Clearly, what Minsky means by "understanding" music or painting is quite different from what, say, Mozart or Picasso meant by the same term. One of his uses of the word "understand" in this quoted passage is essentially a pun—though, I believe, an unconscious one—on the other. The very innocence of his use of it testifies to the tenacity of the hold the metaphor has on him.

This new definition of understanding is now very widely accepted, not only explicitly in scientific circles, but implicitly in the common wisdom. It implies, as the psychologist George A. Miller ruefully pointed out,

> "that the only reason something cannot be done by a universal Turing machine is that we don't understand it. Given this interpretation of what 'understanding' consists of, any attempt to suggest counterexamples becomes merely a confession of ignorance or, if one persists in claiming that he can understand something he cannot describe explicitly, one becomes a prototypical member of that class of people known as mystics."

In other words, the computer metaphor has become another lamppost under whose light, and only under whose light, men seek answers to burning questions.

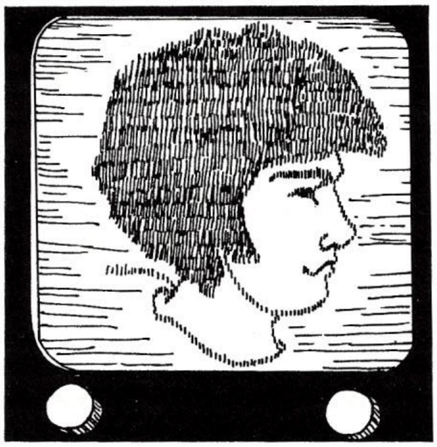
No branch of science has erected this lamppost more deliberately and with more enthusiasm than has psychology. George Miller writes:

> "Many psychologists have come to take for granted in recent years . . . that men and computers are merely two different species of a more abstract genus called 'information processing systems.' The concepts that describe abstract information processing systems must, perforce, describe any particular examples of such systems."

The narrowing of vision that characterizes modern scientific investigation, just like the narrowing of the field of view accomplished by a microscope, can only be justified, and sustained, because it permits us to see things we could otherwise not see. Science and technology have, after all, momentous achievements to their credit. Given the depth to which the computer metaphor has penetrated psychology, it is natural to ask whether it has justified itself there in terms of some tangible achievements.

The impact that the computer, in its role as a high-speed numerical calculator, has had on psychology, although undoubtedly very large, hardly counts, as a "tangible achievement." Psychology has long tried to become "scientific" by imitating that most spectacularly successful science, physics. Psychologists, however, seemed for a very long time to have misunderstood just what it was that made physics somehow more a science than psychology. Like sociology too, psychology mistook the most superficial property of physics, its apparent preoccupation with numbers and mathematical formulas, for the core that makes it a science. Large sections of psychology therefore tried to become as mathematical as possible, to count, to quantify, to identify its numbers with variables (preferably ones having subscripted Greek letters), and to manipulate its new-found variables in systems of equations (preferably differential equations) and in matrices just as the physicists do. The very profusion of energy expended on this program was bound to guarantee that some useful results would be achieved. Psychometrics, for example, is and remains an honorable trade. And there can be no question that statistics benefited enormously from the exercise it was given by psychology, just as it had benefited in the days of its infancy from its application to gambling. Perhaps it repaid its two patrons about equally.

It is often said that the computer is merely a tool. The function of the word "merely" in that statement is to invite the inference that the computer can't be very important in any fundamental sense because tools themselves are not very important. I have argued that tools shape man's imaginative reconstruction of reality and therefore instruct man about his own identity. Yet the folk wisdom that perceives the computer as a basically trivial instrument rests on an accurate insight: the computer, used as a "number-cruncher" (that is, merely as a fast numerical calculator, and it is so used especially in the behavioral sciences), has often, as George Miller has also pointed out, put muscles on analytic techniques that are more powerful than the ideas those techniques enable one to explore. "The methodological rigorists," writes Stanislav Andreski, "are like cooks who would show us all their shiny stoves, mixers, liquidizers and what not, without ever making anything worth eating." The high-speed number-cruncher is, in the hands of many psychologists, merely their newest, shiniest, and most spectacular mixer-liquidizer.



When the computer is used merely as a numerical tool in psychology (or in any other field), it does not usually create a focusing of vision; i.e., it is not comparable to the microscope. It is therefore unrealistic to expect such use to uncover previously unseen worlds or to render distinct what was earlier seen only in vague outline. Because the view of man as a species of the more general genus "information-processing system" does concentrate our attention on one aspect of man, it invites us to cast all his other aspects into the darkness beyond what that view itself illuminates. We are entitled to ask what we would purchase at that cost.

There can be no final answer to such a question, for the extent of the creative analogical reach of a metaphor must, by its very nature, be always surprising and thus not fathomable in advance. We can say in anticipation that the power of a metaphor to yield new insights, depends largely on the richness of the contextual frameworks it fuses, on their potential mutual resonance. How far that potential will be realized depends, of course, on how profoundly the participants in the creative metaphoric act can command both contexts. That is why, for example, the computer expert who knows nothing but computers (the "Fach Idiot," as the Germans call such a person) can derive no broad intellectual nourishment from his expertise and is therefore doomed to remain forever a hacker. That is also why the computer metaphor is, as George Miller puts it, "most productive in areas where a considerable foundation of theory based on previous research already exists."

One area of psychology was extraordinarily well-prepared to benefit from a fusion with the kind of process-oriented thinking characteristic of computer scientists; it was the area which concerns itself with the cognitive processes underlying the acquisition and memorization of information. An enormous amount of laboratory work had been done on, for example, the task of memorizing so-called nonsense syllables. One form of an experiment that has been performed by countless psychological laboratories is to present a subject with, say, a dozen pairs of three-letter syllables, one pair at a time, and to ask him, on each (but the first) presentation of the first of the pair, to say what the second is. The syllables are carefully chosen to be inherently meaningless. Thus, for example, *CAT* is not a nonsense syllable, but *PAG* is. Subjects are exposed to the list, one pair at a time, repeatedly until they are able to give the correct response item to each stimulus item. Edward S. Feigenbaum reported,

> "The phenomena of rote learning are well-studied, stable, and reproducible. For example, in the typical behavioral output of a subject, one finds:
>
> a. Failures to respond to a stimulus are more numerous than overt errors.
>
> b. Overt errors are generally attributable to confusion by the subject between similar stimuli or similar responses.
>
> c. Associations which are given correctly over a number of trials sometimes are then forgotten, only to reappear and later disappear again. This phenomenon has been called oscillation.
>
> d. If a list of x syllables or syllable pairs is learned to the criterion; then a list y is similarly learned; and finally retention of list x is tested; the subject's ability to give the correct x responses is degraded by the interpolated learning. The degradation is called retroactive inhibition. The overt errors made in the retest trial are generally intrusions from the list y. The phenomenon disappears rapidly. Usually after the first retest trial, list x has been relearned back to criterion.
>
> e. As one makes the stimulus syllables more and more similar, learning takes more trials."

The goal of most workers in AI is to build machines that behave intelligently.

Feigenbaum, currently a professor of computer science at Stanford University, conjectured that this sort of learning task involved the subject in an active, complex symbol-manipulation process which could best be described and understood in terms of more elementary symbol-manipulation processes of just the kind that can be programmed for a computer.

Of course, nothing would have been easier than to write a small program for a computer which would have enabled an experimenter to give the computer lists of nonsense syllables that the computer could then reproduce perfectly after the first "trial." The task Feigenbaum set for himself was much harder: to produce, in the form of a computer program, a model of cognitive processes whose over-all behavior would closely approximate that of human subjects engaged in memorizing nonsense syllables, and whose detailed internal functions would constitute a theoretical explanation of the difficulties actually observed in experiments. Moreover, he wished his explanations to be at least consistent with such psychological observations as, for example, that humans have both short-term memories, in which they can apparently hold a few symbols for instant recall during a short period of time, and longer-term memories, in which an almost unlimited amount of information can be stored but from which individual items can be retrieved only at the expense of some effort. If this "effort" to remember is thought of as the computational effort involved in executing a perhaps long subroutine, it becomes obvious how one can begin to apply the computer metaphor.

Feigenbaum's central idea is for the computer to store *descriptions* of the syllables presented to it, not the actual syllables themselves. The syllable *DAX,* for example, may be described by the fact that its first letter has a vertical leading edge and contains a closed loop, that its second letter contains a horizontal middle bar, and so on. When a syllable is first presented to the system, a description of it just sufficiently detailed to allow it to be discriminated from the syllables already stored is added to storage. If it is a stimulus item, that is, the first of a syllable pair, then a "cue" consisting of a minimal description of the syllable with which it is to be associated is stored with it. Because all these descriptions are so minimal, the system often makes wrong associations when presented with stimulus items. But because the correct response item is presented whenever the system makes such an error, the descriptive information then in play may be improved by adding further description to it. Eventually the system learns the list perfectly. When another list is then attempted, the descriptions associated with it may again be confused with those corresponding to the first list, and vice versa. The system is thus capable of exhibiting retroactive inhibition. And clearly, as the items to be learned are made more and more similar to one another, an increasing number of trials is required to refine the discriminating power of each relevant descriptor. The system thus behaves very much as does a human confronted with the same task.

Feigenbaum's program, though by now very old (it was completed in 1959), remains instructive in at least two respects. First, it offers us a relatively simple example of what is meant by a model of a cognitive process in computer-program form. The way it organizes its information storage is meant to be a functional description of the human intermediate-term memory. As such, it explains, for example, how it may be that we can totally forget something for a long time and yet recall it again under certain circumstances. It cannot be that the allegedly forgotten item was simply wiped out of our minds, for it it were, we could never regain access to it. In Feigenbaum's model no information is ever destroyed. But information may be hidden, so to speak, by descriptors leading to other associations; thus one memory may screen or mask another. Sometimes a refinement of the screening image (that is, of a cue) is, in

Feigenbaum's system, all that is required to uncover (that is, to make again accessible) what was previously masked.

Feigenbaum's system also requires that the two syllables to be associated with one another be simultaneously available to the computer (that is, present in its store) for a short time. After a "cue" to the response item has been associated with the description of the stimulus syllable, the two syllables per se can be erased from the computer's store—in other words "forgotten." There is thus a part of his system that plausibly corresponds to what little psychologists know about the function of the human short-term memory. No one, least of all Feigenbaum, claims that his model constitutes "the" explanation for such phenomena. But it is an explanation in a domain where explanations are rare.

The second respect in which Feigenbaum's program is instructive is that it behaves in ways which were not directly and deliberately "programmed in," as the saying goes. For example, the program exhibits what psychologists call interference; that is, the acquisition of a new association interferes with the production of an older one when the syllables involved have closely similar descriptions. The program contains no interference subroutine as such. The phenomenon arises as a consequence of the entire structure of the program, and appeared as a surprise to its designer. In that respect, then, the model *predicted* a behavioral phenomenon, which enormously enhanced its plausibility. The program thus instructs us that the easy and much-repeated slogan "a computer does only what its programmer told it to do" is in certain respects quite wrong and is in any case problematical.

The program we have been discussing is a member of a class of programs called "simulation programs." Their object is to simulate the way humans accomplish certain tasks, but decidedly not to accomplish those tasks in the most efficient way a computer possibly could. We have noted, for example, that a computer could easily be programmed to "memorize" lists of nonsense syllables in one "trial." But that would teach us nothing about how humans might accomplish what appears at least superficially to be the same task.

Because programs which concern the cognitive aspects of human behavior fall naturally within the domain of artificial intelligence, AI (about which more later), they need be distinguished from another class of AI programs, namely, ones that are entirely task-oriented.

Workers in AI tend to think of themselves as working in one of two modes, often called *performance mode* and *simulation mode.* Perhaps the best way to make the distinction clear is by analogy to flying. Virtually all early attempts to understand flying or to build flying models were based on imitating the flight of birds. It is a plausible conjecture that the myth of Icarus, the Greek hero who flew with wings attached to his body by wax and who crashed when the heat of the sun melted the wax, reflects man's early failure to imitate the birds. We might say that the early thinkers and pioneers were operating in simulation mode. Already by the middle of the nineteenth century, however, men like Henson and Stringfellow, and somewhat later Langley, shifted to what we might call performance mode. They considered that their task was to build flying machines based on whatever principles they could discover. Their aim was performance first and understanding only to the extent that it would contribute to performance.

A third mode of operation should perhaps be mentioned in this context: theory mode. There were great aerodynamicists before there were practical aircraft. Lord Rayleigh, for example, published important papers specifically on the theory of flight beginning about 1875. Of course, after the Wright brothers achieved their historic flights in 1903, interest in aerodynamics increased continually and has not flagged to this day. But whereas the aerodynamicist is devoted to theory as such and tends to think of practical aircraft as being mere models of his theories, the aircraft designer looks to theory as being merely another source of ideas which may help him gain more performance from his machines.

The situation in AI is closely analogous to that just described. The goal of a majoritiy of workers in AI is to build machines that behave intelligently, whether or not what they produce sheds any light on human intelligence. They are working in performance mode. They wish to build machines that speak as humans do and that understand human speech, that can, with the aid of television eyes and mechanical arms and hands, screw nuts onto bolts and assemble even more complex mechanical gadgets, that can analyze and synthesize chemical compounds, that can translate natural languages from one to another, that can compose music and complex computer programs, and so on. They are, of course, happy to accept whatever contributions the theoreticians (for example, the psychologists) can make that may help them realize their wishes. But their goal, unlike that of the theoretician who seeks understanding (or claims to), is performance first and last.

A program like Feigenbaum's clearly eschews performance; it is designed to require many trials to learn its lists, whereas, as I have pointed out, if performance were its goal, it could be made to memorize them in one trial.

The dividing line between simulation mode and performance mode is, as might be expected, not absolute. Often the only way to begin thinking about how to get a computer to do a specific task is to ask how people would do it. One thus starts out essentially simulating one's own introspectively observed behavior.

There is, of course, a difference between a program whose avowed aim is performance but that, at least initially, simulates "the way people do it," and a program that simulates what people do in order to learn something about people. But when a program undertaken under the latter banner succeeds, it also performs. Sometimes its authors then cannot resist the temptation to make performance as well as theoretical claims for it, and thus to contribute to the blurring of the line dividing performance mode from simulation mode. Newell, Shaw, and Simon, for example, wrote a program which could prove some theorems in the propositional calculus by simulating the way students who are naive about logic struggled with such proofs. Newell, Shaw, and Simon stated as their aim, "we wish to understand how a mathematician, for example, is able to prove a theorem even though he does not know when he starts how, or if, he is going to succeed." After reporting how long it took their program to prove a number of theorems, they remarked: "One can invent 'automatic' procedures for producing proofs . . . but these turn out to require computing times of the order of thousands of years for the proof of [some particular theorem]." It is hard to read that statement as anything other than a claim that their program can perform usefully, aside from its being possibly instructive about how "mathematicians prove a theorem." As it happened, within a year or two after the appearance of their paper, the mathematician Hao Wang published an "automatic procedure," that is, a computer program, capable of proving all theorems in the propositional calculus. It proved the particular theorem whose proof Newell, Shaw, and Simon estimated would "require computing times of the order of thousands of years" in 1/4 second on what today would be considered a very primitive computer.

The fuzziness of the line dividing simulation made from performance mode is, quite justifiably, a matter of little concern to workers in AI. At the outset of a large research effort, what is important is to have a fairly clear idea of at least the general domain within which questions are to be asked, or, to put it another way, of what it is that is not presently understood that the research is intended to help us understand. Wang's research yielded a result that deepened our understanding of certain aspects of mathematical logic. The aim of the work reported by Newell, Shaw, and Simon was, in their own words, to "understand the complex processes (heuristics) that are effective in problem solving." They chose to examine how people prove theorems merely as an example of human problem solving. Newell and Simon have, as we shall see, pursued their work on problem solving to this very day. What has been problematical about it, and remains so, is in what sense of the word "understand" it helps us to understand man as an information processor or as anything else. That same question can usefully be asked about artificial intelligence generally.

That I have so far cited only very early AI projects is not in any sense "unfair," for AI researchers themselves continue to cite these very examples (namely, Feigenbaum's program, and the logic theory machine of Newell, Shaw, and Simon) as being fundamental work, as far as they go. Newell and Simon have, as I have said, continued their work on problem solving. Meanwhile other workers, notably those at M.I.T.'s and Stanford University's Artificial Intelligence Laboratories have increasingly turned their attention to robotics, that is, to the problems associated with the building of machines that sense aspects of their environments, e.g., with the aid of television eyes, and that are capable of physically acting on it, e.g., by means of computer-controlled mechanical arms and hands. Their work has, as might be expected, generated a host of subproblems in such areas as vision, computer understanding of natural language, and pattern recognition.

Of course, some of these subproblems are also autonomous problems, that is, are independent of the research goals of robotics. Natural-language understanding by computer is an example of a problem that is inherently interesting and difficult in its own right. That any progress on it may prove useful for instructing robots is, to many workers, merely an additional motivation, certainly not the principal one. I shall later have more to say about the manipulation of natural language by computers. For the moment, however, let us turn to some of the more recent work on problem solving, particularly that of Newell and Simon.

The modern literature on problem solving is punctuated by two important books, George Polya's *How to Solve It* and Newell's and Simon's *Human Problem Solving*. Polya's book was first published in 1945, that is, years before electronic computers became practical research instruments. Yet in it Polya lays the groundwork and, in a sense, heralds all the work on problem solving that was to follow for thirty years afterward. Polya's concern is with heuristic problem-solving methods, that is, with those rules of thumb which, when applied, may well lead to a solution of the problem at hand or to some progress toward solving it, but which do not guarantee a solution. Heuristics are thus not algorithms, not effective procedures; they are plausible ways of attacking specific problems. Polya anticipated much of the later work of computer scientists on problem solving when he wrote:

> "Modern heuristic endeavors to understand the process of solving problems, especially the *mental operations typically useful in this process*. . . . Experience in solving problems and experience in watching other people solving problems must be the basis on which heuristic is built. In this study, we should not neglect any sort of problem, and should find out common features in the way of handling all sorts of problems; we should aim at general features, independent of the subject matter of the problem. The study of heuristic has 'practical' aims; a better understanding of the mental operations typically useful in solving problems. . . .
>
> "It is emphasized that all sorts of problems, especially PRACTICAL PROBLEMS, and even PUZZLES, are within the scope of heuristic [sic]. It is also emphasized that infallible RULES OF DISCOVERY are beyond the scope of serious research. Heuristic discusses human behavior in the face of problems. . . . Heuristic aims at generality, at the study of procedures which are indepen-

Heuristics are not algorithms, not effective procedures; they are plausible ways of attacking specific problems.

dent of the subject-matter and apply to all sorts of problems."

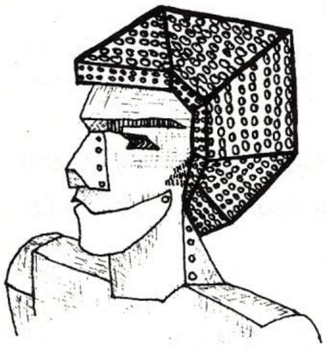
No clearer charter for the work of Newell and Simon could have been written. Polya, in effect, predicted those aspects of what Newell and Simon were later to do which most truly characterize their conception: the endeavor to understand mental operations, the emphasis on generality, on independence from subject matter, and on the usefulness of watching people solve problems, and the stress laid on puzzle-solving behavior. Finally, Polya emphasizes that his book is about methods, and that the most important heuristic is "the end suggests the means."

What Newell and Simon were later to call "the means-ends method" was first suggested when the way an early version of their logic-theory machine proved theorems was compared with recordings of "thinking aloud" sessions of nonmathematics students attempting the same tasks. These so-called *protocols* proved highly suggestive for further work. Protocol taking, that is, watching other people solve problems, became virtually a hallmark of Newell and Simon's procedure.

The new information-processing psychology proceeds from the basic view:

> "that programmed computer and human problem solver are both species belonging to the genus 'Information Processing System' (IPS) . . . .
>
> "When we seek to explain the behavior of human problem solvers (or computers for that matter), we discover that their flexibility—their programmability—is the key to understanding them. Their viability depends upon their being able to behave adaptively in a wide range of environments . . . .
>
> "If we carefully factor out the influences of the task environments from the influences of the underlying hardware components and organization, we reveal the true simplicity of the adaptive system. For, as we have seen, we need postulate only a very simple information processing system in order to account for human problem solving, in such tasks as chess, logic, and cryptarithmetic. The apparently complex behavior of the information processing system in a given environment is produced by the interaction of the demands of that environment with a few basic parameters of system, particulary characteristics of its memories.
>
> "Matters are simple, not because the law of large numbers cancels things out, but because things line up in a means-ends chain in which only the end points count (i.e., equifinality)."

This is a truly remarkable statement, especially in light of Simon's claims that the hypothesis it represents "holds even for the whole man." It behooves us to attempt to understand just what this "very simple" information-processing system is which produces complex behavior as a function of its environment and "a few basic parameters." We must also ask what it is about tasks like chess, logic, and cryptarithmetic that generalizes to the "whole range [of problems] to which the human mind has been applied," that is, to that range to which these same authors have promised computers will be applied "in the visible future." The last question is especially pertinent because existing heuristic problem-solving programs deal only with very simple problems in chess and logic. Cryptarithmetic hardly counts here, since it is what is called, even in AI circles, a "toy problem."



We have already agreed that it is entirely proper and even useful to assume a very particular viewpoint and, from the perspective it affords, to see man as an information processor. And since the computer, the Turing machine, is a universal information processor, it is natural to compare man as seen from that perspective with the computer. Information-processing *psychology* is, however, *not* information-processing *neurophysiology*. It does not attempt explanations in terms of bits or by making analogies to flip-flops, electronic circuits, and so on. It eschews, and rightly so, even explanations that depend on comparison with the sort of symbol manipulations that classical Turing machines do, e.g., writing, reading, erasing, and comparing extremely simple and irreducible symbols such as zeros and ones.

Recall that a program for a particular computer is essentially a description of another computer, that it transforms the former machine into the latter. One can therefore design a computer, and subsequently implement it in the form of a computer program, whose "built-in" elementary information processes (*eip's*—the terminology is Newell and Simon's) are ones that operate on arbitrarily complex symbol structures, that is, that read, write, erase, compare such symbol structures with one another, and so on. Such structures can be made to represent formulas in logic, mathematical expressions, words, sentences, architectural drawings, etc., and, of course, computer programs which may themselves then be manipulated by eip's.

A particularly useful programming device is, for example, to organize information in the form of concatenations of individual items. The "link" that chains one item to the next is a machine address, a pointer, that is stored next to one of the items and points to its successor. A list (as such chains are commonly called) of items so concatenated may then again be considered an item and may thus be pointed to by still another item. In this way, structures of very great complexity may be created and manipulated. Specific eip's may treat them as single items, whereas others may course over them, inserting and deleting substructures, for example.

Information-processing psychology is not information-processing neurophysiology.

An information-processing system is therefore, in this context, a hardware computing system together with a program capable of executing eip's on stored symbol structures. It has, of course, input-output equipment, such as console typewriters, that enable adequate communication with the world outside itself.

The most ambitious information-processing system that has been built for the purpose of studying human problem-solving behavior is Newell and Simon's General Problem Solver (GPS).

"The main methods of GPS jointly embody the heuristic of means-ends analysis.... Means-ends analysis is typified by the following kind of common-sense argument:

> I want to take my son to nursery school. What's the difference between what I have and what I want? One of distance. What changes distance? My automobile. My automobile won't work. What is needed to make it work? A new battery. What has new batteries? An auto repair shop. I want the repair shop to put in a new battery; but the shop doesn't know I need one. What is the difficulty? One of communication. What allows communication? A telephone . . . and so on.

The information-processing theory of man constitutes virtually a dogma for the artificial intelligence community.

This kind of analysis—classifying things in terms of the functions they serve, and oscillating among ends, functions required, and means to perform them—forms the basic system of heuristic of GPS. More precisely, this means-ends system of heuristic assumes the following:

1. If an object is given that is not the desired one, differences will be detectable between the available object and the desired object.
2. Operators affect some features of their operands and leave others unchanged. Hence operators can be characterized by the changes they produce and can be used to try to eliminate differences between the objects to which they are applied and desired objects.
3. If a desired operator is not applicable, it may be profitable to modify its inputs so that it becomes applicable.
4. Some differences will prove more difficult to affect than others. It is profitable, therefore, to try to eliminate 'difficult' differences, even at the cost of introducing new differences of lesser difficulty. This process can be repeated as long as progress is being made toward eliminating the more difficult differences."

To see how this works on one of the kinds of problems to which GPS has actually been applied, consider the following cryptarithmetic puzzle:

$$\begin{array}{r} DO \\ +\,IT \\ \hline TTD \end{array}$$

A subject is told that the above is an encoding of a problem in ordinary addition. Each letter represents a number, and no two letters represent the same number. His task is to assign numbers to the letters in such a way that the given expression represents a correct addition. He is to produce a protocol, that is, to say out loud what he is thinking. Following is one possible such protocol, interspersed with an analysis in GPS-like terms:

Subject: $D+I$ must be greater than 9 because there is a carry to the next column.

Analysis: The subject applied the operator "process column."

Subject: T must be 1 since it is a carry.

Analysis: The subject applied the operator "assign value." He has reached a subgoal and reduced the difference between the given and the desired object. The "given object" is now

$$\begin{array}{r} DO \\ +\,I1 \\ \hline 11D \end{array}$$

Subject: O must be at least 2.

Analysis: The subject applied the operator "generate possible values" to O. (There must have been some unspoken tentative application of the operator "assign value" whose results were rejected.)

Subject: Let's try $O=2$.

Analysis: The subject applied the operator "assign value." Another reduction of difference. The "given object" is now

$$\begin{array}{r} 32 \\ +\,I1 \\ \hline 113 \end{array}$$

Subject: $I=8$.

Analysis: The "assign value" operator is applied and the difference between the given object and the desired object removed. The goal is reached.

This is a much simpler problem than those typically given to subjects and to GPS. A much more typical example of a problem that has been fully analyzed is

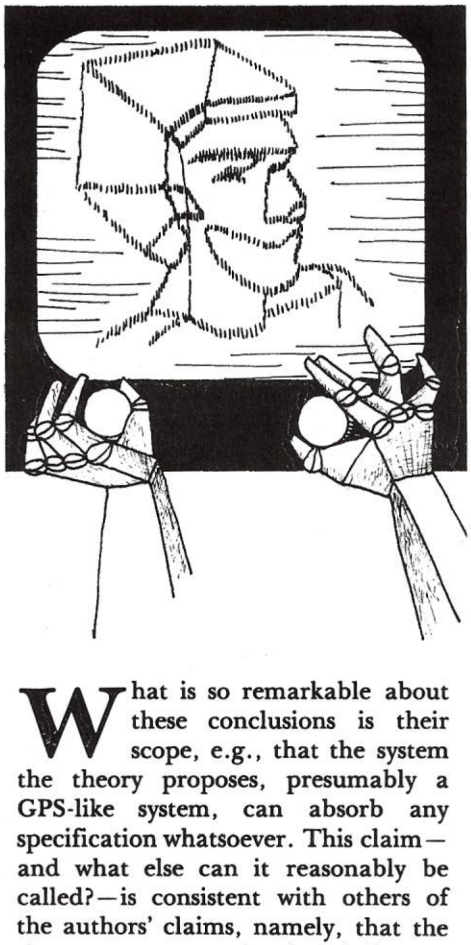
$$\begin{array}{r} DONALD \\ +\,GERALD \\ \hline ROBERT \end{array}$$

where $D=5$. The example we have worked out suffers from the additional fault that it does not display any wrong moves, backtracking, and so on. Nevertheless it gives a general, if pale, idea of the way GPS works and of what a protocol is.

It should also be understood that GPS is not the model of Newell and Simon's theory. GPS implies more about a distinct level of generality independent of the tasks to be accomplished than their theory requires. Indeed, there does not exist any one computer program that is a model of their theory. Instead there exists a number of programs, by no means all of them composed by Newell and Simon or their co-workers, that are substantially consistent with the theory and that employ the "main methods of GPS" listed above. It is the information-processing theory of man which concerns us here, not GPS as such. And we are concerned with that theory precisely because it, in one variation or another, sometimes explicitly and sometimes implicitly, underlies almost all the new information-processing psychology and constitutes virtually a dogma for the artificial-intelligence community.

The basic conclusions the theory reaches are the following:

> "All humans are information processing systems, hence have certain basic organizational features in common; all humans have in common a few universal structural characteristics, such as nearly identical memory parameters. These commonalities produce common characteristics of behavior among all human problem solvers.
>
> "Since the information processing system [i.e., the human seen as an information-processing system—J. W.] can be factored into (1) basic structure, and (2) the contents of long-term memory [i.e., programs and data], it follows that any proposal for commonality among problem solvers not attributable directly to basic structure must be represented as an identity or similarity in the contents of the long-term memories—in the production system or in other stored memory structures.
>
> [The theory] "proposes a system that, given enough time, can absorb any specification whatsoever—can become responsive to the full detail, say, of an encyclopedia (or a library of them). Hence the theory places the determination of differences and similarities of behavior directly upon the causes defining the content that will be stored in the human long-term memory. But these determinants of content are largely contingent upon the detail of the individual's life history. This does not mean that the determining processes are arbitrary or capricious or unlawful. It means that the contents can be as varied as the range of physical, biological, and social phenomena that surround the individual and from which he extracts them."



What is so remarkable about these conclusions is their scope, e.g., that the system the theory proposes, presumably a GPS-like system, can absorb any specification whatsoever. This claim—and what else can it reasonably be called?—is consistent with others of the authors' claims, namely, that the theory can account for the whole man and that computers will, within the visible future, handle problems over the whole range of human thought. The absurdity of what is being claimed for a GPS-like system is underscored by Newell and Simon's assertion that "The apparently complex behavior of the information processing system in a given environment is produced by the interaction of the demands of that environment with a few basic parameters of the system particularly characteristics of its memories." This is of course entirely consistent with their belief that man (like the ant) is "quite simple." But in this context a technical claim is being made, namely, that GPS is quite simple, in the sense that, by changing a *few* of its parameters, its interaction with its environment will produce appropriately varied behavior simulating that of man.

In ordinary technical discussion we speak of a system being sensitive to "a few parameters" when the whole of its relevant mode of behavior can be entirely predetermined by setting a few switches or by entering a few data into its information store. A ship's navigation computer is of this type, for example. It will navigate the ship anywhere given only the geographical coordinates of its destination, some weather data, and so on. But to convert a GPS system from a chess player, say, to a cryptarithmetic puzzle solver is not a matter of changing a few numbers. In effect, the entire "memory structure" of GPS has to be replaced whenever GPS is to switch from one task to another. In other words, GPS is essentially nothing more than a programming language in which it is possible to write programs for certain highly specialized tasks. But, unless a computer program is to be considered a single parameter, GPS does not constitute any support for the claim that the complexity of human behavior is a function of only the human environment and a few parameters internal to the human information-processing system.

Occasionally, Newell and Simon do express a note of caution as when, for example, they admit that "we do not know what part of all human problem-solving activity employs a problem space, but over the range of tasks and individuals we have studied—a broad enough spectrum to make the commonalities nontrivial—a problem space is always used." But then, such mild disclaimers are countered by statements such as, "In spite of the restricted scope of the explicit evidential base of the theory, we will put it forth as a general theory of problem solving, without attempting to assess the boundaries of its applicability," and "we believe that the theory we are putting forth is much broader than the specific data on which we are erecting it."

It is precisely this unwarranted claim to universality that demotes their use of the computer, computing systems, programs, etc., from the status of a scientific theory to that of a metaphor. They themselves say it: "Something ceases to be metaphor when detailed calculations can be made from it; it remains metaphor when it is rich in features in its own right, whose relevance to the object of comparison is problematic." The question then is, can detailed calculations be made from their "theory"? (I shall continue to use the word "theory" here, since it would be too awkward to always write "alleged theory" when referring to the work in question.)

No theory that sidesteps questions can possibly be a theory of human problem solving.

The answer seems, at first glance, to be a resounding "yes." Is not Newell and Simon's book filled with examples of calculations made by GPS? But there is a subtle point here, a point of great importance, a point almost universally overlooked by workers in artificial intelligence who also believe themselves to be in possession of genuine theories. This point is perhaps most clearly illuminated by contrasting Feigenbaum's rote-memory simulator with the GPS programs reported in Newell and Simon's book. Feigenbaum's program is, as I said earlier, a model of a psychological theory, that is, of how people struggle with the task of memorizing nonsense syllables. The program itself is also a theory, as I pointed out; for example, if it is given to a psychologist who is familiar with the programming language in which it is written, one may expect that he will understand it. The property it has which qualifies it as theory, however, is that it enunciates certain principles from which consequences may be drawn. These principles themselves are in computer-program form, and their consequences emerge in the behavior of the program, that is, in the computer's reading of the program. Among them are the well-known phenomena of interference and retroactive inhibition that I mentioned earlier.

The situation is entirely different when, say, the logic-theory program is run in GPS. To be sure, the LOGIC THEORIST is again a theory (albeit a quite trivial one), specifically, a theory of how novices go about solving certain elementary logic problems. But GPS, and this is the crucial point, *is merely a framework within which the logic-theory program runs.* GPS is, in effect, a programming language which it is relatively easy to write logic-theory programs, cryptarithmetic programs, and so on. The elementary information processes, the eip's, which constitute its elementary instructions are simply the primitive instructions of the machine into which GPS has transformed its host computer. GPS as such does not contain any principles—unless one counts as principles such observations as that, to solve problems, one must operate in terms of very general symbolic structures representing objects, operators, features of objects, and differences between objects, that one must build up a library of methods, and so on. Even then, GPS does not permit one to draw consequences from such "principles."

To say that GPS is, in any sense at all, an embodiment of a theory of human problem solving is equivalent to saying that high-school algebra is also such an embodiment. It too is a language, a computing schema, within which one can represent a theory already arrived at by other means. There is, of course, a theory of algebra. And there are theories of programming languages. But neither pretends to say anything about the psychology of human problem solving.

The counterargument to the above thesis is that the theory proposes a system—a GPS-like system—that embodies "a commonality among problem solvers" in its basic structure. It is that basic structure, that embodiment of the commonality among problem solvers, which makes it relatively easy to write problem-solving programs in a variety of quite disparate areas, e.g., logic and cryptarithmetic, in a GPS-like system. But, as Newell and Simon themselves said, any such commonality, if not attributable directly to basic structure, must be represented either in the program written in the GPS formalism or in the stored memory structures. In fact, all current versions of GPS-like systems have such absolutely minimal structure that the information that must be given them (in the form of program and data) for any particular problem-solving task must be detailed and specific, i.e., must define what the relevant operators are, to what objects they may be applied, what "difference" they make when applied to the proper object, and so on. As Newell and Simon say:

> "Due account must be taken of the limitations of GPS's access to the external world. The initial part of the explicit instructions to GPS have been acquired long ago by the human in building up his general vocabulary. This [information] has to be spelled out to GPS."

There, precisely, is where the question is begged. For the real question is, what happens to the whole man as he builds his general vocabulary? How is his perception of what a "problem" is shaped by the experiences that are an integral part of his acquisiton of his vocabulary? How do those experiences shape his perception of what "objects," "operators," "differences," "goals," etc., are relevant to any problem he may be facing? And so on. No theory that sidesteps such questions can possibly be a theory of human problem solving.

The dream of the artificial intelligentsia is to bring into the world "machines that think, that learn, and that create."

The dream of the artificial intelligentsia—a happy phrase the world owes to Dr. Louis Fein—is, of course, to bring into the world "machines that think, that learn, and that create," and whose ability to do these things will increase until "the range of problems they can handle will be coextensive with the range to which the human mind has been applied," as Drs. Newell and Simon already announced in 1958. Their book was published fourteen years later and, as they promised, "the [machines'] ability to do these things increased rapidly," although the then "visible future" appears not to have arrived yet. But the vision is still clear enough. Now, indeed, they have told us how the trick is to be achieved. The proposed system, given enough time (but within the visible future), will become responsive to the full detail of a library of encyclopedias. In order for it to become thus responsive, however, it too will have to acquire a general vocabulary comparable to that commanded by an adult human; it will have to master natural language and internalize a fund of knowledge coextensive with that commanded by the human mind. A large segment of the artificial-intelligence community is, in fact, concentrating on the problem of computer understanding of natural language. That is the problem I intend to discuss in the next chapter.

For the moment, however, it remains to ask what image of man as problem solver can engender—I will not say justify—the mind-boggling vision here presented? To answer that question, we must look, first of all, at what Newell and Simon mean by a "problem."

Newell and Simon write,

> "If we provide a representation for [what is desired, under what conditions, by means of what tools and operations, starting with what initial information, and with what access to resources], and assume that the interpretation of [the symbol structures that represent this information] is implicit in the program of the problem-solving information-processing system, then we have defined a problem."

And then, of course, since "all humans are information-processing systems," one can apply to them and to their affairs the "main methods of GPS," that is, "heuristic means-ends analysis," the testing of objects to see if they are "undesired" and therefore yet to be transformed by operators into the "desired objects," and so on.

It may be objected that such a characterization of the aims of artificial intelligence is a playing with words that unjustly overstates AI's actual and much more modest goals, that Newell and Simon and the AI community generally are really only talking about a certain class of technical problems to which the above definition applies and for which GPS-like methods are surely appropriate. But the point is precisely that the pervasion—we might well say perversion—of everyday thought by the computer metaphor has turned every problem into a technical problem to which the methods here discussed are thought to be appropriate. I shall have more to say on that theme later.

Let it suffice for now to note that H. A. Simon had already written in 1960,

> "Let us suppose that a specific technological development permits the automation of psychiatry itself, so that one psychiatrist can do the work formerly done by ten.... This example will seem entirely fanciful only to persons not aware of some of the research now going on into the possible automation of psychiatric processes."

The research he had in mind was that then just begun by Kenneth Mark Colby, a psychoanalyst, who wrote,

> "Having conducted many laboratory experiments on free-association and having had years of clinical experience with neurotic processes, my initial hope was to simulate both[!] the free-associative thought characteristic of a neurotic process and its changes under the influence of a psychotherapist's interventions."

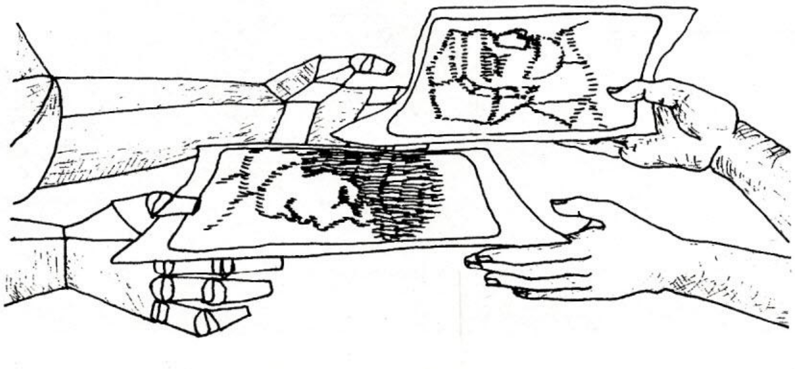
The project—happily—failed. But Simon's words were to ring in Dr. Colby's ears for another six years before emerging again from his own pen. As we have already noted, my own work on the ELIZA system rekindled his enthusiasm and moved him to write the passages I quoted earlier but which bear repetition here:

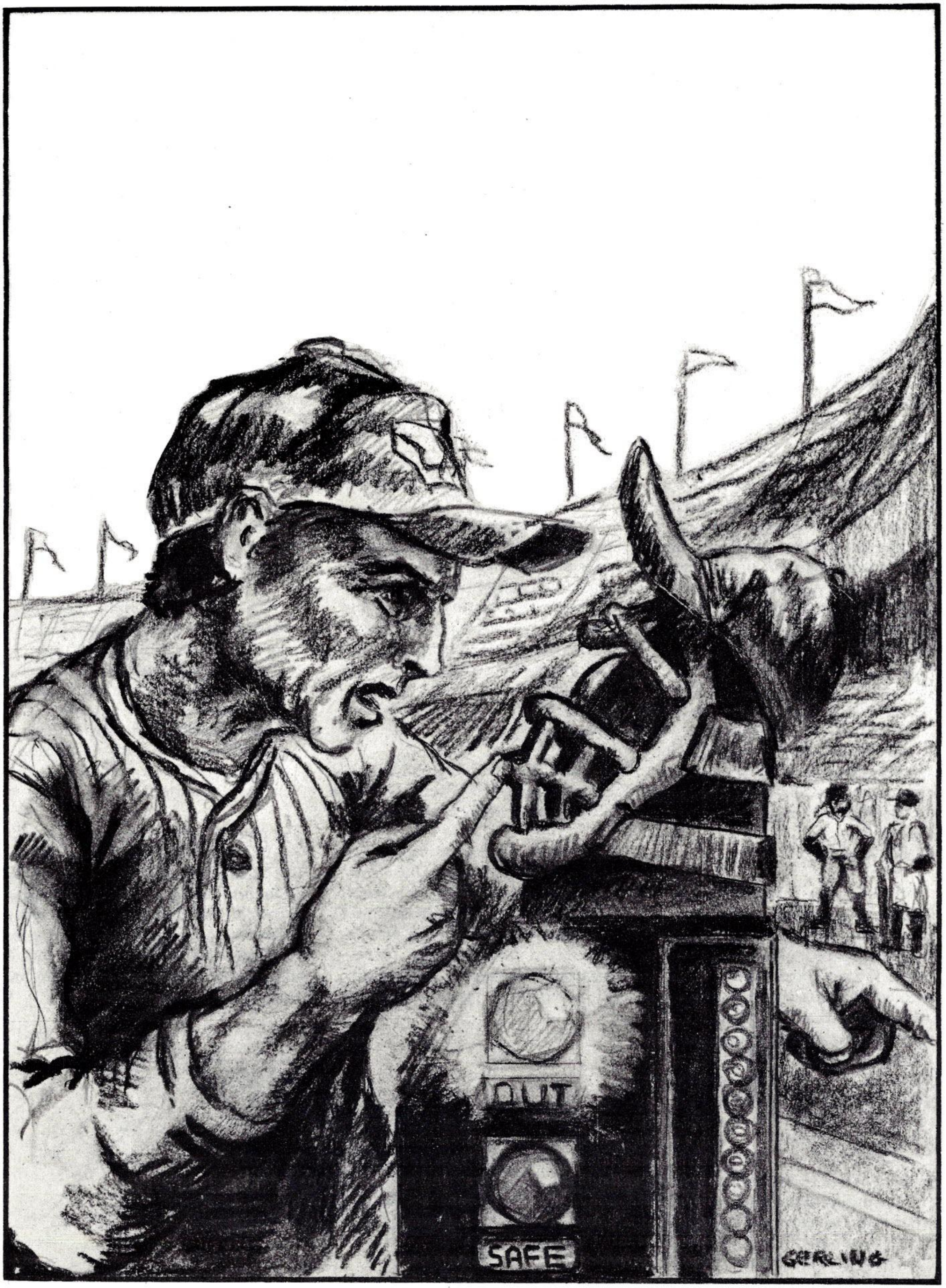
> "If the [ELIZA] method proves beneficial, then it would provide a therapeutic tool which can be made widely available to mental hospitals and psychiatric centers suffering a shortage of therapists.... Several hundred patients an hour could be handled by a computer system."

Just as Simon predicted, and then some! Of course, this euphoric promise is predicated precisely on a view of man as a GPS-like machine. As Dr. Colby said,

> "A human therapist can be viewed as an information processor and decision maker with a set of decision rules which are closely linked to short-range and long-range goals.... He is guided in these decisions by rough empiric rules telling him what is appropriate to say and not to say in certain contexts."

The patient is, in other words, an object different from the desired object. The therapist's task is to detect the difference, using difference-detecting operators, and then to reduce it, using difference-reducing operators, and so on. That is his "problem"! And that is how far the computer metaphor has brought some of us. ▼




OUT
SAFE
GERLING

Fair Play

A Short Story by Steve McCue

Blooper Daly stormed onto the diamond. His sunken blue eyes blazed in a red corona of bursting vessels, his face contorted in a knot of solid rage. It was a familiar scene. Everyone in the league savored Daly's sporadic tirades against calls with which he disagreed. He swiped at the electronic bat boy as he stomped down the first base line. You could swear that steam was rising off his bare, balding head. Yes, this was going to be a good fight.

In their cubicles fans puffed on their risky reeds and woofed down pep corn.

Players of both teams melted away from the area of dispute. Only the impassive ump stood stoically in the path of the oncoming tornado. The play had been the most common in baseball. A slow groundball hit to the shortstop, who had gloved it easily and flipped it over to first. The batter, Los Andiago's star third baseman Juan Lopez, must have eaten an extra dose of soy concentrate for breakfast, for, galloping down the line to first, he added a good ten kilometers to his rated, certified speed. So far as the eye could see, Lopez and the ball had arrived in a dead heat at the base. The ump had called him out by a microsecond, and that had brought sixty-year-old, battling Blooper Daly stomping onto the field.

The All-com cameras zeroed in on the action. The sixty-five reporters covering the game for the newsplastics trotted around to the first base side to view the action. The simulator control booth turned up the crowd roar to stimulate excitement among the millions of Los Andiago and Boswash fans watching on their All-coms. In their cubicles the fans puffed on their risky reeds and woofed down pep corn as the old game took on a new twist. The division championship was getting closer; it was September 3, and each play was important. Boswash's Super Socks and Los Andiago's Fiberglassers were in a tight race for the division lead, and it was almost certain that the winner of the division championship would go on to sweep the series. This was a crucial game for both teams, and Daly knew it.

Blooper placed his pink nose right up to the ump's face. He had lost all control and he spat the words out at his tormentor.

"You rusty son of a dinovender. You measly cousin of a multiplexer. Didn't they change your perceptors for the game? My 'termie' shows that he was safe. He exceeded his rating by twenty percent."

Some ump, just a pile of moon steel and cracked fiber optics!

He placed his stubby finger under the perceptor he was doubting and continued, screaming now:

"You're a refugee from a reclaim pile. Your damn electrons have slowed to Mach 2. Some ump, just a pile of moon steel and cracked fiber optics, that's what you are." He was pushing now with his finger on the placid face in front of him. His rage was still growing.

The battle was in vain, as everyone including Daly realized. The ump's decision was final of course, absolutely correct. Even the closest refiguring never changed their minds, much less showed them to be wrong. But that didn't stop guys like Daly from trying. The league allowed the show of emotion, and even encouraged a bit of violence by letting angry managers off with token fines. The fights weren't the same without the fans in the stadiums to cheer on their side. (Fans hadn't been allowed in the ball parks since the Ankorage massacre ten years before.) But a good smashed auto-ump with dials and gauges scattered all over the field sometimes jumped the All-com ratings for a team fifteen percent. And auto-umps were easily repaired, it was well worth the price.

"Mr. Daly," the auto-ump said in a high pitched voice, "you are testing my core's patience. I have already strained my programs' parameters for dealing with you. Continued dispute of my ruling will bring your ejection from this contest." Auto-ump 336 blinked a light emitting diode as it spoke.

Illustrated by Steve Gerling

Blooper didn't much care. He pushed hard with his finger under the perceptor and hauled his left hand back. During his playing days, several years before, that hand had been able to hurl a ball with such a rated, certified velocity that he had been the most feared southpaw in the game. Now that sixty-year-old left hand, made into a fist, suffered a compound fracture as it crashed into the mass of fused metal and glass. He spent the evening in the emergency room, had to eat with his right hand for two months, couldn't ever again move the little finger in his left hand, and Lopez was still out at first.

A good smashed auto-ump, dials and gauges scattered over the field, jumped the All-com ratings fifteen percent.

The season went on, grinding into September, with Boswash and the Fiberglassers pressing for the division lead. According to all the statistics, it shouldn't even have been close. Los Andiago had the pitching, hitting, and depth on the field to roll over the division. Yet, they couldn't win the close ones. During July, Blooper could have sworn it was the pitching that was off. During August, he thought the hitting had gone to hell. Now, in late September, he just wasn't sure what was costing them the one run ball games. He would pass one excuse a day along to the Archbishop, who owned the team. The Archbishop was only interested in confessions at church. The prelate suggested in rather un-religious language that Daly might like to try his hand at coaching a team in a warmer climate, a much warmer climate.

The press was fuming too. Bunky Ramsdall, the syndicated columnist from the Los Andiago *Surfer,* had blasted him and his team from one end of the column to the other. "Blooper's termie is plugged into a control core for the Los Andiago zoo, and it's making monkeys out of the team." This was a particularly hard jab at Blooper's ego. He prided himself on his use of the termie for managing his team. Analyzing stats had always been one of his strong points.

Fans sent him notes written in mozzarella cheese; they said he would do much better in life running a chain of pizza dispensers. He even got a call from a Congressperson for his block suggesting that the Internal Subcommittee on Organized Inefficiency might be interested in his case. The Archbishop had let it be known through other than clerical circles that old Blooper's head for figures might not be what it used to be. And worst of all, the team's morale was at a subterranean level. They even stopped complaining about the quad-bucks in their contracts. The complaining had switched to old Blooper as the cause of their ills. Blooper was beginning to think that they might be right.

Even his co-hab of twenty-seven years was concerned. One night, after inserting their magnetic stripe grocery card into the dinovend, she had let him know her feelings:

"Knock a few of them around like you used to," she said. "That's what the team needs." She had ordered Blooper's favorite, rhino steak, medium rare. "Mother's in her eighties and she's attached to this town. Moving to another town will, well, who knows? She might have to give up working at the pool, and you know what being a lifeguard means to her."

"Suzie, honest to God, I'm doing my best. I check and double check all the statistics. I get the boys out there for practice evaluation at eight each morning. I've even made some of them drop their advertising contracts. And I know they're trying hard too." He bit into the rhino steak, savoring its delicious flavor. "I just can't put my finger on it. It's a problem I've never run into."

"Maybe I ought to start cutting the menu a bit," Suzie said as she sipped a bit from her glass of synthetic champagne.

"Have faith, my dear; I'll get to the bottom of this. You can bet your pantyhose on it."

She smiled sweetly at him. Her green, tinted hair looked stunning in the dim light of dusk. He loved that little woman. She had spunk. She had stuck with him through it all—through the days in the minors out on Guam and Okinawa, through his first faltering years as a major league player in Alaska, and through thick and thin as he had started his managing career after his playing days were over. The six years at the Institute in Cooperstown had been rugged, studying every night the intricacies of the great system that stored and provided the information which made sports what they were in the nation today. There hadn't been much time for her, but she knew that he did his best and she was with him for the duration. He had studied hard, learning all that was possible about the management section of the Sports Info System. Understanding the four major functions—administration, adjudicating (the auto-umps), playing concepts analysis, and public relations enhancement—was vital in his profession.

Being a manager meant knowing the parameters of the massive system, how to key his terminal into the data base that made successful baseball possible. His greatest difficulty was understanding the software makeup of his sport. The new computer language developed for the game just didn't come naturally to him. Jock II was an intricate, complex programming language that was designed with sports in mind. Today it hardly seemed possible that he could have had trouble in putting a good program into the machine. Many of the programs that he had developed were copyrighted, and some were so secret that only he knew the access codes to them. Squeeze Play IV was one of those. It took the rated and certified speed of the batter and the opposing infield, combined them with the wind velocity, the bunting ability of the hitter, plus the altitude of the stadium, and arrived at the standard deviation from the chance of getting a run in by the old squeeze play. The programming for the play was kept in locked vaults in Cooperstown and could only be tapped by Blooper with the Los Andiago code. Like all other information, it was used in real time, over finely conditioned transmission lines which were constantly checked for leakage.

Daly remembered his graduation day. His professor had taken him down to the massive memory banks of the system. They had stood there under glaring fluorescent lights and stared at the machine as it quietly handled the constant flow of requests. His professor had placed his hand on the smooth metal wall. Turning his crinkled face to him, he said, "She'll serve you well, Blooper. She won't let you down. This baby has the knowledge to take you to the top. She'll take you all the way. Have faith in her boy."

Blooper had never forgotten those words. While some managers didn't bother to double check their stats on simple plays, Daly faithfully consulted his termie, for every situation. When a pitcher had been banged around for five runs in one inning, or when a player hadn't hit for seven games straight, he still checked with his termie to find the odds on removing him from the game. He counted on that "baby" and she hadn't let him down yet, that is, until now.

They finished supper. Suzie said she had to go over to her mother's to do calisthenics. Blooper headed for the living room to relive some plays with the system. He was going to try to get a handle on the problem with the team.

He closed the bubble on the reliver and punched the authorization code on his termie; the digits shot out at the speed of light over the fiber optic "wires," screaming to the core of the Cooperstown computer. Instantly he was back in the dugout, reliving Lopez' out at first. First came the call; then the surge of anger as he stood before the auto-ump. He heard the high pitched voice, and again he splattered the ump's face across the field. The machine cut out just as the pain began. He was sitting once again in his living room.

He lit a risky reed and considered the play.

A thought gnawed at his mind, just below the surface of his consciousness. If he could only bring it to the surface, his problems would be over. The team could win; he was sure of it.

The Archbishop suggested in un-religious language that Daly might try coaching in a much *warmer climate.*

Suzie came home at ten-thirty. He was dozing, still inside the bubble. She saw by the setting on the termie that he had been working. Well, that was better than a lot of husbands who used the reliver when their co-hab was out to visit a red light computer bank. Good old Blooper, she wouldn't tell him what mother had said about his managing the team. She helped him into their bed. Legs entwined, they fell asleep.

At four that morning Blooper sat straight up in bed, his eyes wide open. He reached under the pillow for his termie and punched up the system in Cooperstown. He delved into the logic test center and it confirmed his thoughts. He was sure he had come upon the answer to his problem. The Fiberglassers would win. He awakened Suzie, who listened in a stupor to his idea.

"I've got it, Suzie. I knew it would come to me."

"Yes, dear, I'll call mother in the morning and tell her not to put her cubicle up for barter. Now go back to sleep. You have a big game tomorrow."

She stroked his head and now, his problem solved, he fell back to sleep. It was only natural, he thought, just like Suzie's mother. You had to have a bat to play ball. He would place the necessary order in the morning.

Blooper stood in front of the team in a pregame meeting in the locker room. The boys were in various states of dress, punching their gloves, knocking their water spikes on the benches, nervously awaiting the beginning of the crucial game.

"OK, men, listen up," Blooper said. "Today's our day; I mean it. We're going to get out there and beat those guys." The team had heard that every day for the last month, but today there was something different in his tone. Maybe it was an edge of confidence, or an inflection of courage. Homer Smith, the all-star first baseman, put down his copy of the newsplastic's franchise page and listened. Ziggy Lacy raised his hand.

"Boss, what's going to make this game any different?"

"Ziggy, it's simple. We've been taking the wrong approach toward the auto-umps. We've got to start being nice to them. No more spitting out your nicotine gum onto their perceptors. Quit bouncing line drives off them during batting practice. Harry, you've got to stop letting the pitchers bean them during pitching practice. And we all, including me, have got to stop taking swings at them." Blooper tried to drive home his point with a quick smile. There was an incredulous look on their faces. Be nice to auto-umps? This team had prided itself on its creative ability to send them to the repair shops and here was Blooper, who had punched more auto-umps than anybody, telling his team to be nice.

"Ah, come on Blooper, be nice to auto-umps?" The new rightfielder, just traded with Capetown, broke in.

"Yea, be nice, real nice, and there's another reason, too. Sawdust, come on in here; I want the boys to meet you," Blooper said as he turned to the door to watch the player he had sent for this morning enter the room.

Smitty dropped his dome glasses. Lopez punched for his mitt, missed, and knocked over a Super Coke next to the Capetown rightfielder. Homer pulled up his pants.

"Boys, meet Sawdust Olender, our new weapon," Blooper said to the shocked Los Andiago Fiberglassers.

The Super Socks took a quick two to nothing lead in the top of the second. Daly was a bit annoyed by the call that had allowed the second run to come home. Once again his termie was telling him that the rated and certified speeds of his rightfielder and the speed of the Boswash runner should have had him out. Yet the auto-ump had called him safe. Yesterday he would have jumped onto the field ready to do battle. Today he smiled sweetly at the damn machine and was happy to see his catcher apologize for swearing at the play. That took a lot of control for old Bunky. He made a mental note to cut him in on his soyburger franchise in Spokatle.

The boys were doing just great, tipping their hats to the umps, smiling at them and even expressing concern about the first base perceptor that was smashed by a foul ball. He congratulated the boys between innings. At least they were with that part of the program. About Sawdust? Well, he didn't know. Olender was seated in the corner of the dugout, avoiding the curious glances of the players. Daly was saving his secret weapon until just the right moment. Not only would it handle the auto-ump problem, but he would go down in baseball history. His name would be among the great managers of the game: McGraw, Stengel, O'Malley, and now Daly.

Los Andiago managed to tie the contest in the bottom of the eighth with a long drive by Homer into the plexiglass shield in far left. Boswash couldn't score in the top of the ninth and Blooper decided to pull out the stops and use Sawdust after two were down in the ninth. A hush descended on the dugout as Sawdust donned the hitter's helmet and strode to the plate, swinging a thirty-ounce bat. The Boswash manager sprang to his feet; he checked his termie and found that Sawdust was on the Fiberglassers' roster. Jeers and groans emerged from the Boswash dugout. The newsplastic reporters sprang to their feet muttering among themselves. The All-com cameras zoomed in on the player walking to the plate. Fans from all over the nation sat up and watched as history was made before their eyes.

As the Super Socks' manager screamed at the home plate auto-ump, Daly calmly walked across the field, gave Sawdust a pat, and said, "Don't worry kid, he doesn't have a chance. You're gonna get up to bat."

The auto-ump stared impassively at the two managers, squared off in front of the green and red perceptors of the home plate model.

"Daly, this is the lowest blow of all. You can't win fairly so you decide to put up this, this...!" The Boswash manager was pushing on Daly's chest.

"Now look, old buddy, you're offending this good ump's ears with your words. My goodness, you know there's no rule against using Sawdust." Daly was calm, collected. The players of both sides were yelling insults at each other. Daly was glad to see that his team backed him in this matter too. He wondered if there were enough soyburger franchises available for the whole team. They all deserved it.

The auto-ump spoke, "Please return to your respective dugouts." The high pitched voice galled Daly, but his smile didn't fade. "Player Olender can bat as an authorized pinch hitter."

The Boswash manager looked at Daly contemptuously. "Well, my pitcher's just gonna throw that old horse hide so fast Sawdust won't even see it. We all know one of them can't hit. I don't care how well they do with their own kind. This here is the big leagues."

Daly sat down calmly. He waited for the pitch. He hoped he was right. He rechecked his termie; the stats looked good. It was between Sawdust and the auto-ump now.

Grovie Peters wasn't the meanest pitcher around, but the sight of Sawdust at the plate turned on his mean genes. He wound up and threw a brusher ball that nearly clipped off the batter's nose. The next pitch was aimed at the gut. Now with two balls and Sawdust brushed back from the plate, he aimed the next screeching fastball right down the middle. The blue light for a ball shone on the auto-ump's metal forehead. Grovie was mad, but waved the catcher back behind the plate and threw another one right down the middle. Again the blue light. Sawdust took first base. The catcher turned and screamed at the auto-ump. The Boswash manager stomped onto the field. Blooper just sat in the dugout with a satisfied look on his face.

The argument was on, and no one noticed that the orange time-out light had not lit on top of the auto-ump's head, no one but Sawdust, that is. The Los Andiago player was on third by the time Boswash realized it. The auto-ump would not listen to any of the Boswash protests.

"My program states that players may advance on base if play is not called. Neither contesting teams nor auto-ump officials did so; therefore, the player is advanced to third base," sing-songed the ump. Blooper watched the Boswash manager smash the home plate ump to bits of cracked metal with one of the bats. He yelled out across the field, "Tut, tut, old man, let's get on with the game." A new ump was brought out. The manager was ejected and the game continued.

Jock II was an intricate, complex programming language designed with sports in mind.

Blooper sent Homer to the plate. He had checked his termie for the odds on a squeeze play. With two out in the ninth and Sawdust on third with only a mediocre rated speed, the odds were impossible. But, well, there were other factors to be considered. He signaled a squeeze.

Homer took the first pitch waist high for a strike. Sawdust was taking a big lead at third. Homer dropped the next pitch for a perfect bunt down the first base line. The pitcher rushed in, gripped the ball barehanded and drilled it home. The catcher had the ball a full step ahead of Sawdust diving for the plate.

The newspeople crowded around Blooper after the game. He stood with his arm around Sawdust on the steps of the dugout. A broad smile covered his face as he answered the questions posed by the reporters. A young man with a handheld microcamera pushed past the rest. With a touch of sarcasm in his voice, he asked:

"Blooper, how do you explain that final call at home plate? Replays show that Sawdust was a foot behind the ball. It sure looked to us up in the booth that the call should have been out."

"Hey, buddy, you know these babies don't change their calls. I mean the whole Sports Info System is behind them. All I can say is that the catcher must have missed the tag. It happens to the best of them, you know."

The reporters left. Blooper and Sawdust walked slowly out onto the field.

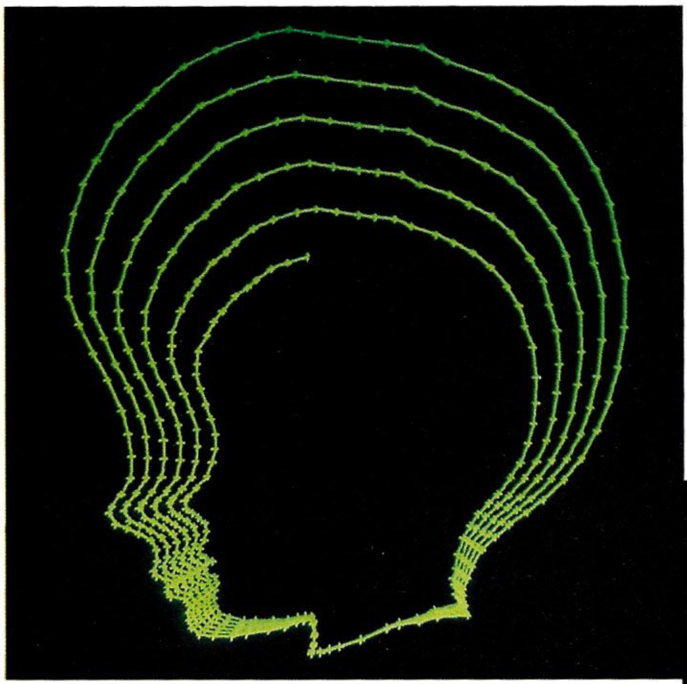
"You're gonna be a great asset to the team. By the way, I've managed to get you a private shower. It took some doing, but the upper bosses finally bought it," Blooper said.

They stood on home plate as the sun, setting over the edge of the stadium, spewed its filtered rays onto them. The talk diode lit on the top of the auto-ump.

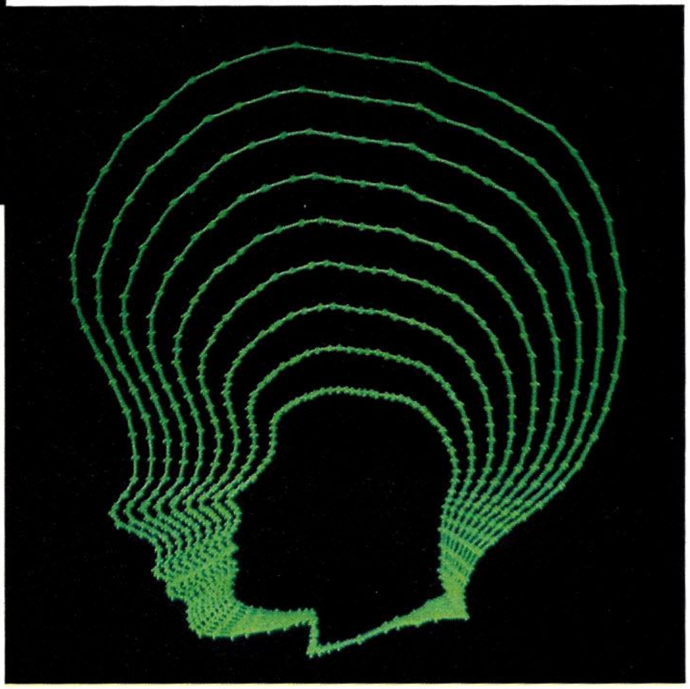
"If I were you, dear," the ump said in its high pitched voice, "I'd get a hairdresser written into my contract too."

Blooper cracked a smile, "You would say that, you girls always stick together."

Sawdust slapped Blooper fondly on the rear. "Hey, Blooper, after all it's only fair play, isn't it?" ▼



Micro, micro on the wall, How will I look when I am tall?



by Stuart Dambrot

Imagine the shock of recognition at seeing yourself as you will look twenty years from now. If the reality of that classic sci-fi dream, the time machine, is not here—indeed, may well never be here—then, its rudimentary sibling, the time-image projector, has arrived. Seeing one's future self, at least a shadow of one's future self, is no longer a fantasy.

Robert Shaw, a professor of psychology at the University of Connecticut, who has been delving into effects of growth and aging on the shape and appearance of the human head and the way in which we perceive these changes, has developed a computer system that can actually take an image of your head and project it five, ten, even twenty years into the future. You look at yourself as you will be—assuming, of course, you survive that long. (That's a factor over which his computer has no control.)

Dr. Shaw and his fellow researchers are studying both the nature of craniofacial transformations and their relationship to biological "age judgments," that is, our impressions of a person as young, old, or somewhere in between. Their work, known as Project FATE (an acronym for Facial Aging and cosmetological Treatment Effects), is a computer-oriented program designed to be utilized in orthodontics and in medical treatment of head and face disfigurements.

Project FATE is based upon a mathematical procedure or transformation. Shaw developed the transformation for measuring and predicting changes in surface, shape, and proportions that occur in the human skull as an individual grows up and then old. The next step was to write a computer program that applied the aging transformation to a two-dimensional profile of a generalized human skull. The

Photographed by Thomas Hall

PORTRAIT SCULPTURE BY

The complexities of sculpting the heads used in the FATE Project have been greatly simplified by some clever computer applications. Foremost among them are those invented by Dynell Electronics Corporation. Dynell operates a Studio for Solid Photography in New York City. There they utilize a process including high-speed optical scanning, advanced electronics, and sophisticated computer technology to transform an ordinary photograph into an accurate and realistic Solid Photograph or "portrait sculpture," of a person or object. Sizes range from one twenty-fourth scale to full size, and the sculptures are accurate to within one-fortieth (.025) of an inch.

Solid Photography had its origin in the kind of intuitive flash one finds throughout the history of science. Paul DiMatteo, president of Dynell, saw the potential for creating solid portraits one morning as he watched shafts of sunlight stream through his bedroom window. After five years and three million dollars spent on research and development, his idea became a reality.

In the first step of Dynell's process, photographs are taken in a studio in back of the showroom. The subject is first seated (or the object placed) in the center of several pieces of modified thirty-five millimeter photographic equipment. There are eight cameras for upper and lower views of four sections of the head; overlapping fields of view assure complete coverage. The room is then darkened. While the exposure is being made, patterns of light from four projectors illuminate the subject's head.

In the photographic process itself a pattern of lines is projected onto the subject. This pattern of lines is coded on film where it appears in the form of line smears. The key to the process is that these distortions of the projected lines are proportional to the shape of the photographed object, thus constituting data from which the computer can generate three-dimensional coordinates describing the object. The line smears themselves are translated into digital signals by an optical scanning device and stored on reels of magnetic tape until they can be processed.

The actual sculpting (using a material called Paralene) occurs in three steps. First, a computer-directed one-bit drill coarse replicator machine cuts sections from a cylindrical block of Paralene, leaving a crude approximation of the final piece. The coarse-cut sculpture is then placed on a fine-cut replicator for the detailed sculpting. The fine-cut machine uses twenty computer-controlled rotating blades, ten on each side of the occasionally rotating coarse-cut sculpture. The result of the fine cutting, in the case of Project FATE, is an accurate, highly detailed reproduction of the subject's head and face.

At this point, a skilled artist does the final touch-up work, refining very small details and perhaps adding a smooth high-luster facial finish. If a cast sculpture is to be made, the Paralene bust is coated with a rubber compound to permit construction of the required mold. The entire process—sculpting, finishing, and casting—takes but a few hours to perform.

Solid Photography has a number of potential applications in the industrial, commercial, artistic, medical, dental, educational, scientific, and military fields. The Dynell studio has already been used for making facial models of dental patients under orthodontic treatment, models for sculpting parts of the human anatomy in radiation therapy and preoperative plastic surgery reviews, and, in a wholly different area, variously scaled models of plastic bottles manufactured by a major container company. This last example illustrates a feature of the Dynell process that is available as a result of the fact that Solid Photography is a computer-based system: a sculpture may be scaled both in a linear and non-linear fashion. The former simply allows a sculpture to be enlarged or reduced in size. Non-linear scaling, on the other hand, lets the programmer/sculptor selectively and precisely change the

---

program worked. When they ran it in one direction, it produced a profile with features characteristic of a young child. Run in the opposite direction, the program generated an "older" profile.

Shaw and his associates verified the validity of the program in controlled experiments where people were asked to rank profiles from youngest to oldest. Their judgments about biological age level agreed with the predictions made by the aging transformation. Not only could experimental subjects correctly identify human facial profiles according to age, they could also recognize young and old cars and profiles of dogs from computer-generated drawings. The transformation, then, provides an abstract model for dynamic "growth" space. It seems that any object, human or not, can be "remodeled."

After verifying its validity, the aging transformation was tested against actual changes in specific human skulls. The researchers obtained thirty-five years' worth of accumulated growth data (x-rays and matching photographs) on 100 individuals from Denver Growth Council studies. From each individual's data they made a skull tracing of that person at a specific age. This tracing was then "transformed," with the computer generating a predicted profile for, say, five years after the original. The computer profile was compared to a second, actual skull tracing of that individual five years later. Using another computer procedure called "pattern matching," Shaw discovered that the accuracy of his transformation program in projections from the original tracing to the target profile of a later date was about

COMPUTER

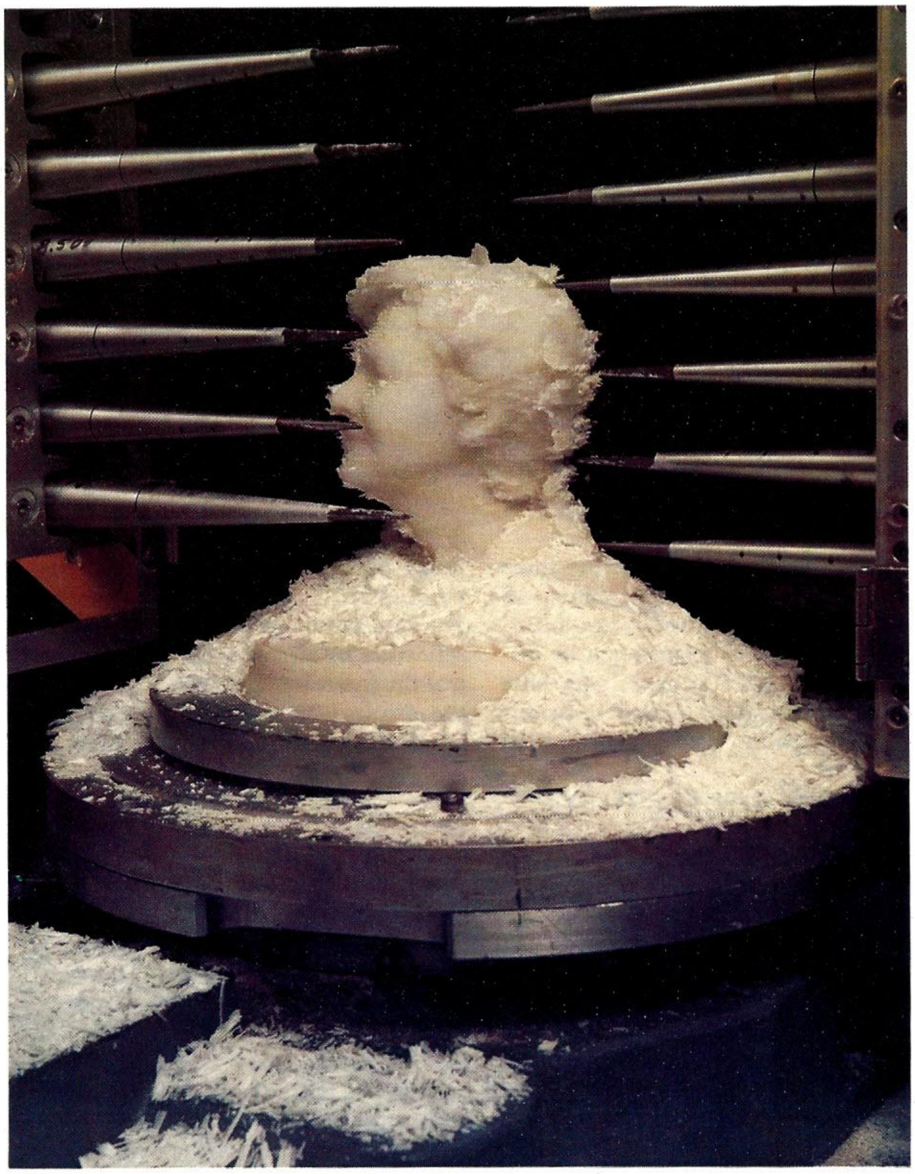
scaling factors of different portions of the object. With the bottles, it is possible, for instance, to keep the size of the handle constant while altering the size of the rest of the model being studied.

Potential applications in the field of artificial-limb construction are yet to be explored. For example, a person missing his right arm could have his left arm photographed at a Solid Photography studio. The data generated by the computer program could then be "reversed" so that a right arm could be sculpted.

A compact, mobile unit could be used by law-enforcement agencies in victim identification: by applying certain to-be-worked-out rules to a Dynell photograph of a decomposed or damaged skull, a bust of the individual as he/she appeared while living could be created.

With a little imagination, other uses become apparent. Imagine an extremely accurate micro studio that could photograph minute objects. Couple this with a solid-state video system. What do you get? Perhaps a means of monitoring and sculpting diminutive biological organisms, such as one-celled plants and animals. A revolution in the field of electron microscopy would be just around the bend.

Meanwhile, Solid Photography is here. A process in search of applications—probably more than we can think of at the moment.

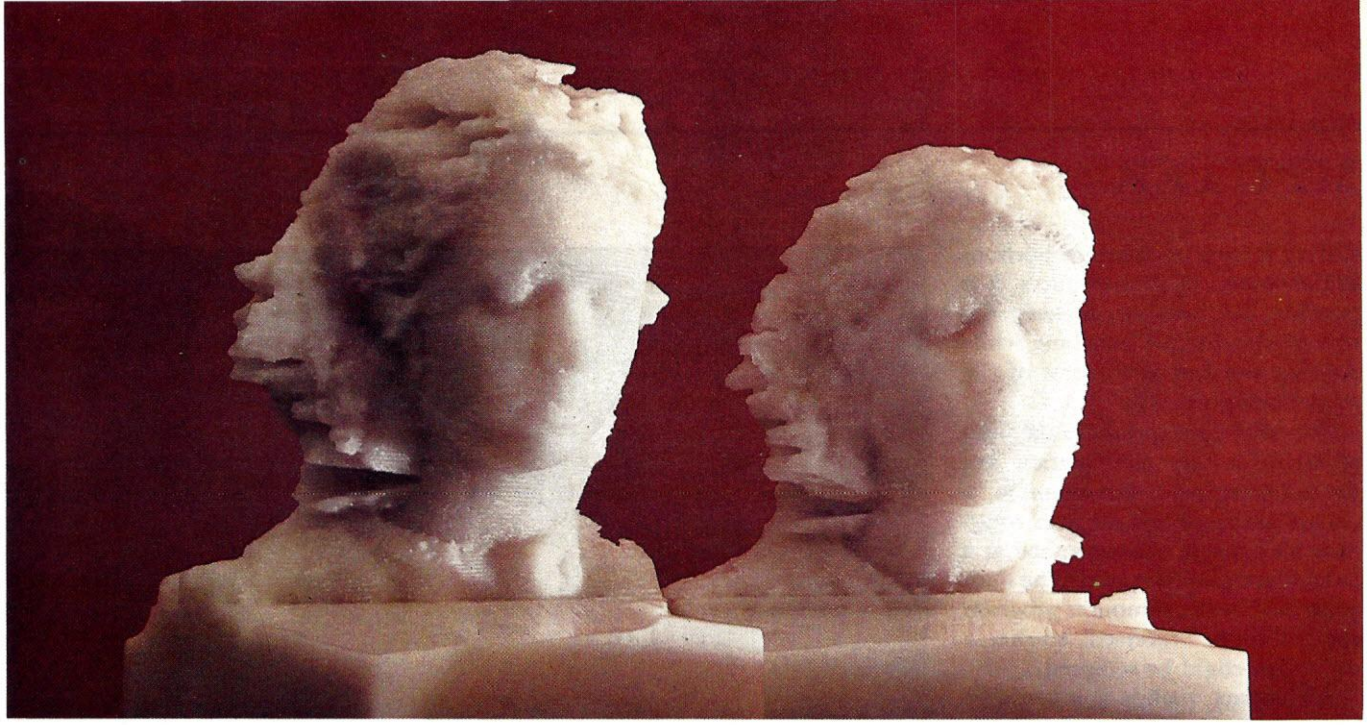


seventy-five percent, a high correlation considering certain unavoidable sources of error, like facial disfigurement resulting from accidents. After all, the computer is not about to predict that you'll break your nose playing football two years from now.

The most striking visual verification of the transformation, however, occurred when Shaw took his fifteen-year-old daughter, Dana, to Dynell's Solid Photography Studio in New York City. There, after Dana's picture was taken, Shaw applied his transformation to the Dynell process, this time reversing the time frame, going back into the past. The Dynell computers produced a bust of Dana as she appeared at the age of four. Shaw's theory, which had succeeded in two dimensions, could be generalized to three dimensions. This was a great step forward.

Currently FATE's programs are being further fine tuned in the computer lab in the University of Connecticut's psychology building by James Todd, the FATE researcher who wrote the original program implementing Shaw's theories. The basic hardware presently consists of a Data General Nova II microcomputer, which has a 32K core memory and which interfaces with a Diablo fixed-head disk, a Kennedy tape drive, a Hazeltine CRT terminal, a graphics tablet, and a Tektronix 611. With this equipment Todd is producing some pretty sophisticated graphics.

The graphics utilized permit the study of changes in human heads. Various subroutines allow the user to input points (through the graphics tablet), store these points (in core or on disk), generate figures—in this case



Dana at fifteen, Dana at four.

profiles of heads—by connecting these points, and manipulate the points and figures in a variety of ways. For example, as many as ten figures can be displayed on the CRT at once, and these figures can be altered either individually or as a group. Another subroutine enables the user to superimpose figures. This is important because it facilitates "fitting" studies in which differences between any two heads or profiles can be precisely measured.

The capabilities of the program are greatly enhanced by the fact that it is interactive; that is, you can "play" with the figures because instructions and data can be entered (via the keyboard or graphics tablet) as soon as the results of the previous transformation are displayed.

One of the FATE programs now in the planning stage is an interesting technique called *shaded graphics,* which is based on a process developed by Dynell Electronics Corporation. Shaded graphics takes a two-dimensionally displayed figure and divides it into a mosaic of triangles. (The display is a raster-scan device; it is similar to a CRT but allows the brightness of each point to be controlled.) Since light shining on an object is reflected at different angles depending on the curvature of the subject, the intensity of the light at different points on an object's surface appears to vary. The brightest spots, for example, are those parts of the surface that are parallel to the observer. Shaded graphics simulates this process of light reflection by allowing the user to control the intensity of different points on the display. In other words, shaded graphics permits the examination of three-dimensional objects on a video screen device. Presto, you'll age right before your eyes on 3-D TV.

You'll age right before your eyes on 3-D TV!

One of the key components for such a three-dimensional display is the cardioid transformation, which is essential to Shaw's theory. Basically, a cardioid, when it is rotated in a complete circle, results in a cardioidal shell, a three-dimesional form that roughly resembles an apple. According to the research, each individual skull has a cardioidal shell that is the "best fit" for that head, and malformations of the face and head can be described as differences between the idealized shell and the actual skull.

If Project FATE's upcoming research is as successful as its past findings have been, then practical applications to various kinds of head and face surgery could be truly spectacular. Facial surgery on children is now a very difficult proposition. Since a child's skull has not completed growing, a false estimation of the direction in which development will occur can exacerbate rather than alleviate the problem. With a projection of the child's future growth pattern to work from, the task becomes not only easier, but the possibilities of success are also far greater.

Consider a specific problem. A doctor must treat a protruding lower jaw, a typical cranio-facial abnormality. This condition can result from an oversized jaw. In this case, the normal treatment involves removal of bone. However, the same condition can also result from an abnormality in the middle portion of the skull and in this case an entirely different type of surgery would be required. What makes the diagnosis and treatment of these problems difficult is the fact that there is usually no way to determine the exact source of the condition.

But imagine an interactive computer system that could utilize the aging transformation, shaded graphics, and Dynell's three-dimensional data-acquisition process. With this setup a plastic surgeon could first have the program generate a display of the patient's head. Through input devices, a mock surgery could be performed on the display and checked against the aging transformation. If the surgical technique was inappropriate, the structural problems that it could cause later in the patient's life would be immediately apparent. The mock surgery could be repeated until the best surgical procedure was found.

As important as mock surgery will be, it is only the beginning. Ponder for a while on what other fantastic applications of this system lie in the future of man—FATE knows. ▼

Copycat Computer

by Tom Digate

This handy file-copy program runs in the Sol System RAM and requires no external RAM. The program may be altered to adjust the speed of each tape unit by changing the values stored at location CA66H for the read unit and CA67H for the write unit. The definitions of the bits in these locations are:

Bit 5 speed; 0 = 1200, 1 = 300
Bit 6 set to 1 for tape unit #2
Bit 7 set to 1 for tape unit #1

Example:

CA66H DB 40H read from tape unit #2 at 1200
CA67H DB 80H write to tape unit #1 at 1200

If a tape read error occurs, the program will print

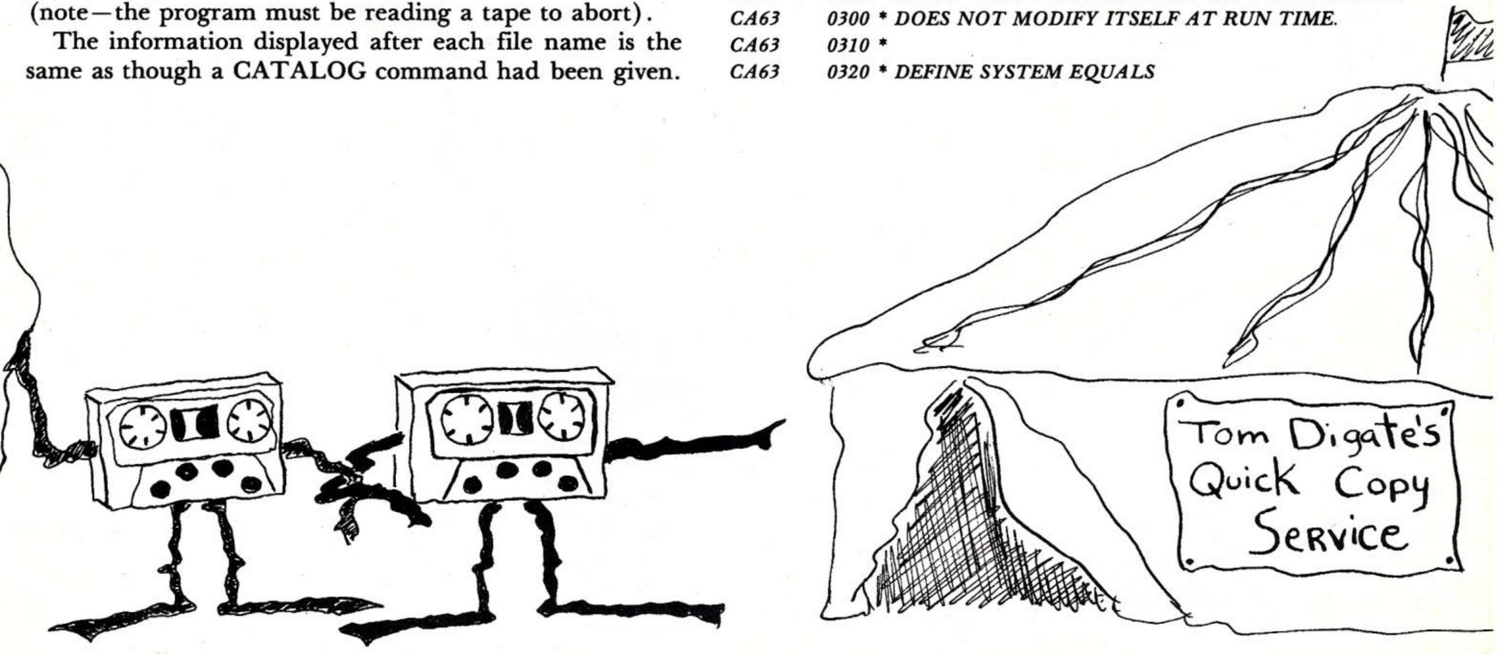
ERROR 'AAAA'

where 'AAAA' is either an old file name or a new file name from the tape. If an old file name is displayed, the program could not read a header correctly. If a new name is displayed, the named program had a checksum error. In the first case, just restart the copy program. In the second case, rewind the tape and retry the read.

To stop the program press MODE SELECT. The program will print an error message and return to SOLOS (note—the program must be reading a tape to abort).

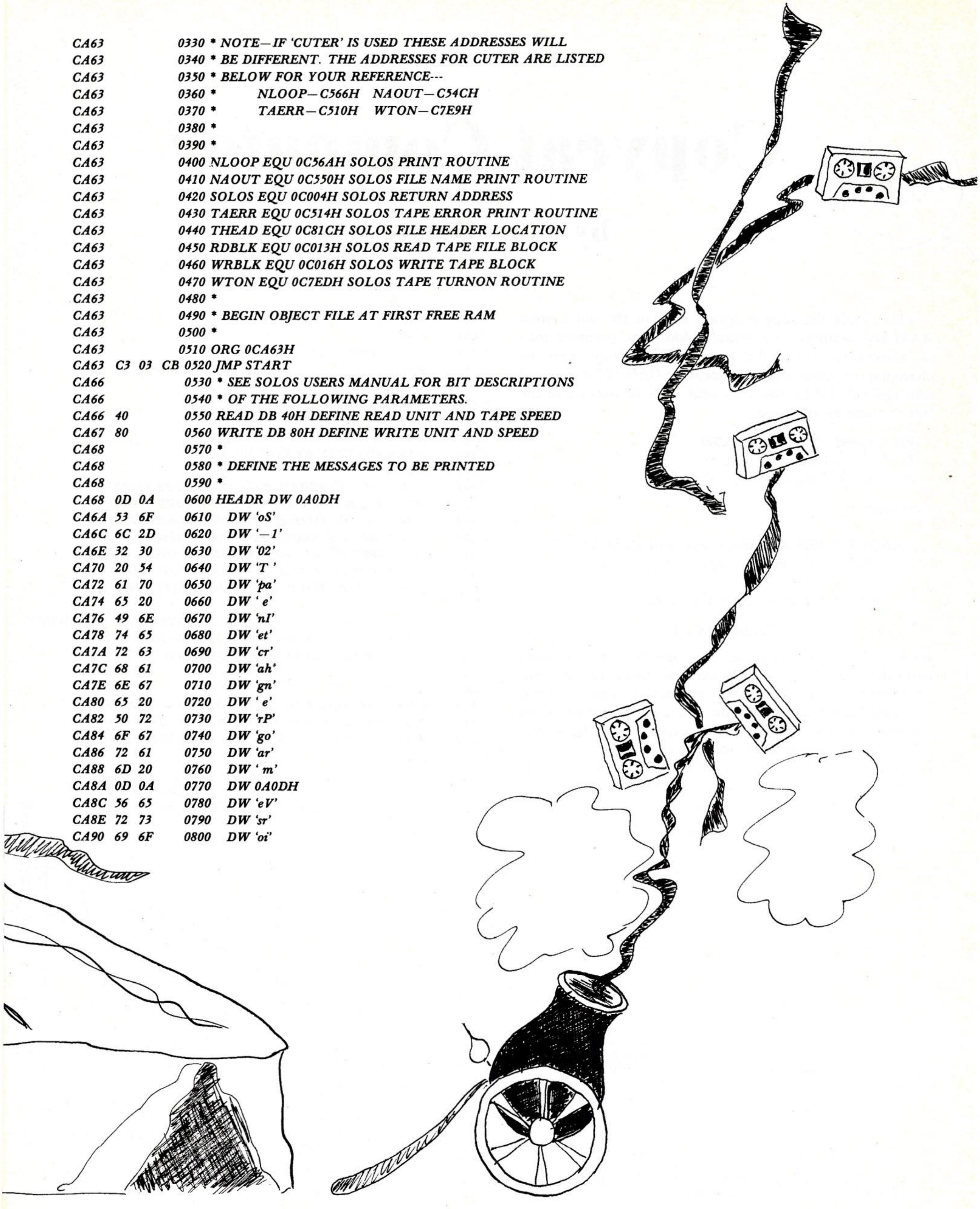
The information displayed after each file name is the same as though a CATALOG command had been given.

```
CA63       0010 * * * * * * * * * * * * *
CA63       0020 *                        *
CA63       0030 * COPY PROGRAM *
CA63       0040 *                        *
CA63       0050 * * * * * * * * * * * * *
CA63       0060 * VERSION 3.0
CA63       0070 * AUGUST, 1977
CA63       0080 *
CA63       0090 * WRITTEN BY TOM DIGATE
CA63       0100 *
CA63       0110 * THIS PROGRAM WILL COPY TAPE FILES
CA63       0120 * FROM ONE TAPE UNIT AND SAVE THEM
CA63       0130 * ON THE OTHER UNIT. NO OPERATOR
CA63       0140 * INPUT IS REQUIRED OTHER THAN
CA63       0150 * SPECIFYING THE TAPE SPEED AND
CA63       0160 * WHICH UNIT IS TO BE THE READ UNIT
CA63       0170 * AND WHICH IS TO BE THE WRITE UNIT.
CA63       0180 *
CA63       0190 * THE PROGRAM WILL STOP AUTOMATICALLY WHEN
CA63       0200 * IT READS AND SAVES A FILE WITH THE FIRST
CA63       0210 * THREE CHARACTERS EQUAL TO "END".
CA63       0220 *
CA63       0230 *
CA63       0240 * THE PROGRAM IS NOW SET UP TO READ
CA63       0250 * FROM TAPE UNIT 2 AND SAVE ON UNIT 1.
CA63       0260 * THE TAPE SPEED IS 1200 BAUD FOR BOTH
CA63       0270 * UNITS.
CA63       0280 *
CA63       0290 * NOTE—THIS PROGRAM CAN COPY ITSELF AS IT
CA63       0300 * DOES NOT MODIFY ITSELF AT RUN TIME.
CA63       0310 *
CA63       0320 * DEFINE SYSTEM EQUALS
```



Illustrated by Korkie

```
CA63            0330 * NOTE—IF 'CUTER' IS USED THESE ADDRESSES WILL
CA63            0340 * BE DIFFERENT. THE ADDRESSES FOR CUTER ARE LISTED
CA63            0350 * BELOW FOR YOUR REFERENCE---
CA63            0360 *        NLOOP—C566H  NAOUT—C54CH
CA63            0370 *        TAERR—C510H   WTON—C7E9H
CA63            0380 *
CA63            0390 *
CA63            0400 NLOOP EQU 0C56AH SOLOS PRINT ROUTINE
CA63            0410 NAOUT EQU 0C550H SOLOS FILE NAME PRINT ROUTINE
CA63            0420 SOLOS EQU 0C004H SOLOS RETURN ADDRESS
CA63            0430 TAERR EQU 0C514H SOLOS TAPE ERROR PRINT ROUTINE
CA63            0440 THEAD EQU 0C81CH SOLOS FILE HEADER LOCATION
CA63            0450 RDBLK EQU 0C013H SOLOS READ TAPE FILE BLOCK
CA63            0460 WRBLK EQU 0C016H SOLOS WRITE TAPE BLOCK
CA63            0470 WTON EQU 0C7EDH SOLOS TAPE TURNON ROUTINE
CA63            0480 *
CA63            0490 * BEGIN OBJECT FILE AT FIRST FREE RAM
CA63            0500 *
CA63            0510 ORG 0CA63H
CA63  C3  03  CB 0520 JMP START
CA66            0530 * SEE SOLOS USERS MANUAL FOR BIT DESCRIPTIONS
CA66            0540 * OF THE FOLLOWING PARAMETERS.
CA66  40        0550 READ DB 40H DEFINE READ UNIT AND TAPE SPEED
CA67  80        0560 WRITE DB 80H DEFINE WRITE UNIT AND SPEED
CA68            0570 *
CA68            0580 * DEFINE THE MESSAGES TO BE PRINTED
CA68            0590 *
CA68  0D  0A    0600 HEADR DW 0A0DH
CA6A  53  6F    0610   DW 'oS'
CA6C  6C  2D    0620   DW '-l'
CA6E  32  30    0630   DW '02'
CA70  20  54    0640   DW 'T '
CA72  61  70    0650   DW 'pa'
CA74  65  20    0660   DW ' e'
CA76  49  6E    0670   DW 'nI'
CA78  74  65    0680   DW 'et'
CA7A  72  63    0690   DW 'cr'
CA7C  68  61    0700   DW 'ah'
CA7E  6E  67    0710   DW 'gn'
CA80  65  20    0720   DW ' e'
CA82  50  72    0730   DW 'rP'
CA84  6F  67    0740   DW 'go'
CA86  72  61    0750   DW 'ar'
CA88  6D  20    0760   DW ' m'
CA8A  0D  0A    0770   DW 0A0DH
CA8C  56  65    0780   DW 'eV'
CA8E  72  73    0790   DW 'sr'
CA90  69  6F    0800   DW 'oi'
```



```
CA92  6E 20      0810   DW ‘ n’
CA94  33 2E      0820   DW ‘.3’
CA96  30 20      0830   DW ‘ 0’
CA98  41 75      0840   DW ‘uA’
CA9A  67 2E      0850   DW ‘.g’
CA9C  2C 31      0860   DW ‘1,’
CA9E  39 37      0870   DW ‘79’
CAA0  37 20      0880   DW ‘ 7’
CAA2  0D 0A      0890   DW 0A0DH
CAA4  57 72      0900   DW ‘rW’
CAA6  69 74      0910   DW ‘ti’
CAA8  74 65      0920   DW ‘et’
CAAA  6E 20      0930   DW ‘ n’
CAAC  42 79      0940   DW ‘yB’
CAAE  20 54      0950   DW ‘T ’
CAB0  6F 6D      0960   DW ‘mo’
CAB2  20 44      0970   DW ‘D ’
CAB4  69 67      0980   DW ‘gi’
CAB6  61 74      0990   DW ‘ta’
CAB8  65 20      1000   DW ‘ e’
CABA  0D 0A      1010   DW 0A0DH
CABC             1020 E1 EQU $
CABC  47 65      1030 GET DW ‘eG’
CABE  74 20      1040   DW ‘ t’
CAC0  2D 20      1050   DW ‘ —’
CAC2             1060 E2 EQU $
CAC2             1070 SAVE EQU $
CAC2  93 97      1080   DW 9793H
CAC4  93 93      1090   DW 9393H
CAC6  93 93      1100   DW 9393H
CAC8  93 93      1110   DW 9393H
CACA  93 93      1120   DW 9393H
CACC  93 93      1130   DW 9393H
CACE  93 93      1140   DW 9393H
CAD0  93 93      1150   DW 9393H
CAD2  93 93      1160   DW 9393H
CAD4  93 93      1170   DW 9393H
CAD6  93 93      1180   DW 9393H
CAD8  93 93      1190   DW 9393H
CADA  93 93      1200   DW 9393H
CADC  2A 93      1210   DW 932AH
CADE  53 61      1220   DW ‘aS’
CAE0  76 69      1230   DW ‘iv’
CAE2  6E 67      1240   DW ‘gn’
CAE4  20 69      1250   DW ‘i ’
CAE6  6E 20      1260   DW ‘ n’
CAE8  70 72      1270   DW ‘rp’
CAEA  6F 63      1280   DW ‘co’
CAEC  65 73      1290   DW ‘se’
CAEE  73 20      1300   DW ‘ s’
CAF0  2A 00      1310   DW 2AH
CAF2  0D 0A      1320   DW 0A0DH
CAF4             1330 E3 EQU $
CAF4             1340 COPY EQU $
CAF4  2A 20      1350   DW ‘ *’
CAF6  43 6F      1360   DW ‘oC’
CAF8  70 79      1370   DW ‘yp’
CAFA  20 64      1380   DW ‘d ’
CAFC  6F 6E      1390   DW ‘no’
CAFE  65 20      1400   DW ‘ e’
CB00  2A         1410   DB ‘*’
CB01  0D 0A      1420   DW 0A0DH
CB03             1430 E4 EQU $
CB03             1440 *
CB03             1450 * PROGRAM BEGINS HERE
CB03             1460 *
CB03             1470 START EQU $
CB03             1480 *
CB03             1490 * PRINT PROGRAM HEADER
```

```
CB03  21 68 CA  1500 LXI H,HEADR
CB06  16 54     1510 MVI D,E1—HEADR
CB08  CD 6A C5  1520 CALL NLOOP
CB0B            1530 *
CB0B            1540 * PRINT ‘GET —’ MESSAGE
CB0B  21 BC CA  1550 NEXT LXI H,GET
CB0E  16 06     1560 MVI D,E2—GET
CB10  CD 6A C5  1570 CALL NLOOP
CB13            1580 *
CB13            1590 * SETUP READ TAPE PARAMETERS
CB13            1600 *  HL = TAPE HEADER ADDRESS
CB13            1610 *  DE = LOAD ADDRESS SET TO SAME AS ON FILE
CB13            1620 *  A = TAPE UNIT AND SPEED
CB13  21 1C C8  1630 LXI H,THEAD
CB16  11 00 00  1640 LXI D,0
CB19  3A 66 CA  1650 LDA READ
CB1C  CD 13 C0  1660 CALL RDBLK
CB1F  DA 14 C5  1670 JC TAERR JUMP TO ERROR ROUTINE IF CARRY SET
CB22            1680 *
CB22            1690 * PRINT FILE NAME AND HEADER INFO.
CB22  CD 50 C5  1700 CALL NAOUT
CB25            1710 *
CB25            1720 * PRINT ‘SAVING IN PROCESS’
CB25  21 C2 CA  1730 LXI H,SAVE
CB28  16 32     1740 MVI D,E3—SAVE
CB2A  CD 6A C5  1750 CALL NLOOP
CB2D            1760 *
CB2D            1770 * GENERATE A LEADER BETWEEN FILES
CB2D  CD 5F CB  1780 CALL LEADR
CB30            1790 *
CB30            1800 * SETUP FILE HEADER ADDRESS IN HL REGISTER
CB30            1810 * AND SETUP TAPE UNIT IN A
CB30  21 1C C8  1820 LXI H,THEAD
CB33  3A 67 CA  1830 LDA WRITE
CB36  CD 16 C0  1840 CALL WRBLK
CB39            1850 *
CB39            1860 * PREPARE TO COMPARE FILE NAME WITH ‘END’
CB39            1870 * ABORT IF EQUAL TO ‘END’.
CB39  21 1C C8  1880 LXI H,THEAD
CB3C  11 50 CB  1890 LXI D,END
CB3F  CD 53 CB  1900 CALL DHCMP
CB42  C2 0B CB  1910 JNZ NEXT
CB45            1920 *
CB45            1930 * PRINT ‘COPY DONE’ MESSAGE AND
CB45            1940 * RETURN TO SOLOS.
CB45  21 F4 CA  1950 LXI H,COPY
CB48  16 0F     1960 MVI D,E4—COPY
CB4A  CD 6A C5  1970 CALL NLOOP
CB4D  C3 04 C0  1980 JMP SOLOS
CB50            1990 *
CB50  45 4E     2000 END DW ‘NE’
CB52  44        2010   DB ‘D’
CB53            2020 *
CB53            2030 * SUBROUTINE TO COMPARE TWO FILE NAMES.
CB53            2040 DHCMP EQU $
CB53  06 03     2050 MVI B,3
CB55  1A        2060 DHLOP LDAX D
CB56  BE        2070 CMP M
CB57  C0        2080 RNZ
CB58  05        2090 DCR B
CB59  C8        2100 RZ
CB5A  23        2110 INX H
CB5B  13        2120 INX D
CB5C  C3 55 CB  2130 JMP DHLOP
CB5F            2140 *
CB5F            2150 * TURN ON TAPE EARLY TO PROVIDE LEADER.
CB5F            2160 LEADR EQU $
CB5F  3A 67 CA  2170 LDA WRITE
CB62  C3 ED C7  2180 JMP WTON WRITE LEADER
```

TALK IS CHEAP

by Hesh Wiener

When interactive computer systems were first developed, they were described as conversational systems because the exchange of data between the user and the computer, via a terminal, was reminiscent of a conversation. Today we are on the threshold of real conversational systems. Computers are beginning to talk and to listen. What's more, most of the devices that make such interaction possible are within the reach of the serious amateur.

While the performance of the most elementary speech synthesizers is limited, the prices of these units are so low that they are hard to resist. Five hundred dollars will get your computer to talk. Programming a system to produce decent speech is a little tricky, but the main ingredients in your success will be patience and insight, virtues commonly ascribed to personal-computing buffs.

Only one voice-input unit I've heard of is suitable for the hobbyist, but it is affordable. With a vocabulary of sixty-four words, which you can readily change, your computer can handle an impressive array of applications.

Other recognition units will be on the market soon—one in its preparatory stages was shown at the June National Computer Conference in Dallas. And with several additional firms supposedly working on speech-recognition devices, the range and quality of these machines should improve steadily during the coming months.

The kind of circuitry used in conversational peripherals is not quite the same as that in your computer. While the speech-input units store their vocabularies in digital form and use digital techniques to match spoken words against stored ones, much of the work is done by analog circuits—circuits that transform currents or voltages rather than representations of numbers. Similarly, the speech synthesizers accept commands from a computer in digital form, but they generate the components of speech with analog circuits.

These machines, which actually have more in common with your stereo than with your computer, represent a major change in the world of personal computing. Because devices that synthesize or accept voiced expression are, for the most part, brand new and very inexpensive, they have placed the hobbyist on a par with computer scientists in the commercial and academic worlds. While expensive speech synthesizers have been available for a few years, and speech-recognition equipment has been developed on a custom basis by several laboratories, the actual number of installations with such equipment has been quite small. Now, however, with prices greatly cut, applications for speech processing will

ANALOG COMPUTERS

In the old days there were analog computers. These machines used the magnitudes and directions of electric currents to represent physical quantities. Input consisted of settings on precision potentiometers, and output frequently consisted of meter readings.

When digital circuits first appeared, they were very expensive, so engineers found ways to get currents to multiply, add, and what-have-you. The problem with these machines was twofold: they were set up by hand and hence slow to program, and their accuracy extended only to three digits, much like that of a slide rule.

As digital computers became less expensive, they forced analog machines into near extinction; however, some problems can still be run more economically on analog computers, particularly if the analog machines themselves are controlled by digital processors. Computers that have both digital and analog calculating elements are called hybrid computers.

With the advent of vocal input and output devices, the world of computing may have come full circle. Once again, it is becoming economical and useful for computer scientists to explore the possibilities of circuits that work with quantities of current rather than representations of numbers.

Of course, advances in circuits that process continuous range signals have not been halted by the inattention of engineering types. Rather, such work has thrived, resulting in such products as CB radios, vastly improved audio and video equipment, heart pacemakers, to name but a few.

The fundamental building blocks of today's linear circuits, as the circuits that transform analog signals are called, are amplifiers, oscillators, and phase-locked loops, circuits that are capable of detecting small changes in frequency. Complex circuits are often built of only a few fundamental elements, much as digital circuits are.

explode all around us. Further progress in the engineering of these systems will probably occur at current price levels, which means that you can keep up with the pros.

Personal-computing buffs will face challenges on several fronts as they get into the world of speech input and output, and their problems will be compounded by the small capacities of their computer systems. This may be a good thing. In the past, limited systems have bred innovation in computing. Examples abound in the early history of mainframes and in the recent history of minicomputers. It would be fairly reasonable, then, to expect the simultaneous efforts of hundreds of personal-computing pioneers to produce elegant solutions to the problems inherent in small speech systems.

BASIC SPOKEN HERE

SPOKEN WORD	STANDARD BASIC	DICTIONARY ENTRY NUMBER
zero-nine	0-9	1-10
point	.	11
plus	+	12
minus	—	13
times	*	14
divide	÷	15
power	↑	16
open	(	17
close	)	18
equal	=	19
let	LET	20
print	PRINT	21
go	GO TO	22
read	READ	23
if(. . .go)	IF. . .THEN	24
speech	SPEECH [1]	25
loop. . .limit. . .step	FOR. . .TO. . .STEP	26-28
root	SQR	29
sine	SIN	30
cose	COS	31
tangent	TAN	32
next	NEXT	33
quote	“	34
run	RUN	35
break	break key	36
list	LIST	37
blank	blank character	38
a-z	A-Z	39-64

---

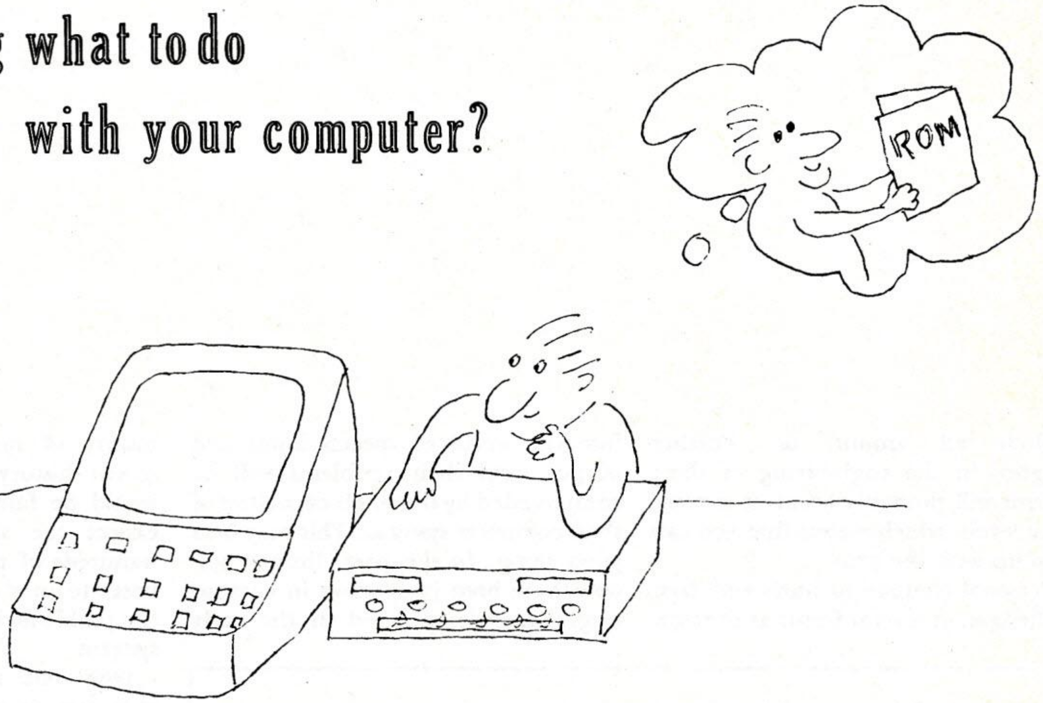
1. SPEECH is the function that calls the recognition unit.

Only one speech-recognition machine now on the market is suitable for the amateur, the SpeechLab, built by Heuristics, Inc. It is a complete hardware and software system, and it even includes a microphone. According to the people who build it, the system is ninety-five percent accurate. The vocabulary of the SpeechLab is limited to sixty-four words. And with sixty-four bytes of storage per spoken word, you won't want to spend too much time with the old tongue twisters such as "antidisestablishmentarianism." The SpeechLab uses the memory and processor of your computer to do its digital processing, so you may want to begin by teaching it only a few things.

The SpeechLab's own language is a version of Tiny BASIC called SpeechBASIC that lets you work in a higher-level language while the computer does all the tough stuff. This is more than I expected when I first heard of the gadget, and it will mean a lot to people who are more interested in results than the art of pattern recognition. In addition to the basic board and microphone, you will also get a handbook that, according to Heuristic's John Rekjallin, "will get you up to 1970" along the paths taken by the discipline's pioneers, if you want to learn about the nitty-gritty.

Whether or not you become fascinated by the technical side of speech recognition, you will eventually begin to play around with applications for the machine. While a sixty-four-word

Wondering what to do with your computer?



ROM will show you.

New ideas, applications, peripherals, games, and just plain good reading about computers and the world from the best writers in the field every month.

Subscribe now and save.

ROM
COMPUTER APPLICATIONS FOR LIVING

ROM Publications Corp.
Route 97
Hampton, CT 06247

Name
Address
City
State Zip

☐One year $15 ☐Two years $28 ☐Three years $39
☐Check or money order enclosed.
☐Master Charge ☐BankAmericard
Exp. date Card#

° Please allow 4-6 weeks for delivery.

GUARANTEE

If not satisfied with **ROM** at any time, let us know. We'll cancel your subscription and mail you a full refund on all copies still due you.

vocabulary doesn't seem large, it is really quite adequate. Suppose you wanted to get your computer, now equipped with a speech recognition unit, appropriate software, and enough memory, to accept and run BASIC programs dictated to it. The vocabulary you would need for a decent BASIC language, suitably modified to remove all homonyms, is actually only—you guessed it—sixty-four words.

A much smaller vocabulary will suffice for most other applications. A program to play chess, for example, could well be written with an input vocabulary of fewer than sixty-four words. You will need six for the chessmen, two for the colors, eight for the rows of the board, numbers, and a few special terms like "checkmate."

Tic-tac-toe on a numbered game board takes nine words for the positions, two for the players, plus "start" and "stop."

More complex applications require a speech synthesizer as well as a speech recognizer. A speech synthesizer is not the only way to give a voice to a computer, but it is the most flexible way. The computer industry has built voice-response units for years. The early ones used sounds recorded on film or—rarely—magnetic tape to provide the computer with a vocabulary. Circuitry in the unit selects and plays the appropriate track.

Recently, the industry has developed similar systems that store the sound digitally in a read-only memory (or programmable read-only memory) and play it back much the same way. The remaining circuits in these machines select and play back the appropriate pre-recorded sound.

The vocabulary of a voice-response unit, unlike that of a speech synthesizer, is fixed. However, the quality of the recorded voice is excellent and, particularly with the all-solid-state machines, it always sounds exactly the same.

To synthesize speech, electronic circuits must produce sound components similar to those produced by the human vocal tract. These sounds must then be combined, much as they are mixed in the cavaties of the throat, mouth, and nose. Finally, a speech synthesizer should offer some way to add anthropomorphic qualities to the sound. For although a great range of expressiveness is not necessary for intelligibility, expression makes the synthesizer sound more human, or at least less mechanical.

While speech synthesizers are able to duplicate some of the functions of the human vocal tract, not all can be imitated. The sound waves that come from the synthesizers are produced by oscillators and sources of so-called white noise that are far simpler than the organic sources of sound we use to speak, sing, whistle, whisper, and yodel. Amazingly, these crude circuits work pretty well.

When speech synthesizers accept as input the names of various phonetic sounds produced by the human vocal apparatus, electronic circuits produce approximations of the sounds called for. Because speech synthesizers respond to the names of (or symbols for) phonemes rather than the letters of the word being synthesized, the data used to represent speech is different from the data used to display a word on the screen of a video display terminal.

At first the phonetics will seem awkward, but it really doesn't take long to catch on. What will take a little longer is adding the touches that make a synthesized word come alive. That takes time.

When you first write a program that talks, it is far easier to test it at a terminal than it is on the actual synthesizer. Once the logic of the program checks out, you can translate the PRINT statements into commands for the synthesizer. Alternatively, you might want to get the output down pat

PHONEMES OF AMERICAN ENGLISH

The people at AI Cybernetic Systems, who build the Model 1000 speech synthesizer, have written some very interesting material about phonics. Even I could understand it. Here is the gist of what they have said.

Modern American English has grown from several different languages. Along the way, the relationship between the letters we write and the sounds we make with our voices has broken down. As a result, to write out American English phonetically is similar to, but not identical to, writing our alphabet.

We use many vowel sounds, most of which can be represented by a single letter pronounced in a certain way because of its environment or because of the derivation of the word it is in. There are the old familiar long and short sounds of *a, e, i, o,* and *u;* there is the sound of *y* at the end of a word and in some other places, plus *er* and a bunch of others. Each of these sounds has a unique representation in the phonetic alphabet, and must have a unique representation for programming on a speech synthesizer. Those unique representations are sometimes called *phonemes.*

Consonants are a little easier than vowels, although some of them, like the sound of *th,* change from word to word. (Listen to the sound of *this* as opposed to the sound in *thin.*) Consonants are produced by the motion of air, and sometimes by the motion of air plus the vibrations of vocal cords. For example, the sound of *p* is unvoiced; the sound of *b* is voiced. Both are called plosives or stops because they are created by stopping the flow of air for a moment and then letting it explode. *T* and *d* are plosives formed by the tip of the tongue near the teeth. There are also sibilants (like soft *s*) and similarly formed sounds that are voiced, like that of a *z.* Sounds like *f* and *v,* unvoiced and voiced, are fricatives. Fricatives are made by constricting the vocal passage. *M* and *n* are nasals and *r* and *l* are laterals. You also have to know about glottal stops, momentary halts in your expiration controlled, logically enough, by your glottis, that little thing hanging from the back of your mouth. (Some people say the word *bottle* with a glottal stop in the middle instead of a *t* sound.)

Now, after all this, can you tell a vowel from a consonant? Well, a vowel is just a noise you make with your vocal cords letting air pass relatively freely from the lungs. Consonants are made by constricting or stopping the passage of the air. The question of whether a *w* as in *woe* or a hard *y* as in *you* are vowels or consonants is a toughie, like whether a tomato is a fruit or a vegetable. I can't quite remember the answer to either question, but I know them when I see (or hear) them. The people who make speech synthesizers know them too.

and then write the program. What is important is keeping chaos from your computer by not trying to do everything at once. Speech synthesizers and speech recognizers are like any other addition to your personal computer—you have to master the hardware before you can really do interesting things with the system.

Once your computer can talk and listen, you will undoubtedly have more ideas than you could ever follow through. The best ones will make your machine seem suddenly to come alive after months of confinement in its cabinet.

Your newly adapted computer can help with cooking if it's near the kitchen or if you can get your kitchen wired up to your audio devices. Cooking involves the control of heat over time, so you will need a way to keep track of time. You will also have to bank some data that is now in the software manual for your stove—your cookbook—as well as the phonetic representations of the appropriate words.

Once you have the basics down, you can solve the big problem—writing a program that will help you keep track of the things you are cooking while you watch TV, debug another program, ice your cake, or lapse into absentmindedness.

A "cook's helper" program can cue you to begin cooking each dish at the right time. Even though each food requires a different cooking time, all of them have to be ready at mealtime. Your data is the length of time it takes to cook each item, and the key datum is the length of time it takes to cook the dish requiring the longest cooking time.

Suppose you are making roast chicken which takes, say, two hours. Maybe your other dishes are rice, which takes twenty minutes to cook after the water has been heating for seven minutes, and carrots, which steam for fifteen minutes after the steamer has heated up for ten minutes. Tell all that to the computer.

When you light the oven, tell your system. Fifteen minutes later, when the oven is hot, your machine should say something like, "Put in the chicken." It will continue to repeat this message until you put the chicken into the oven and tell the computer to start the show. At appropriate times your system should tell you to boil water for the rice, then for the carrots, to put the vegetables in, to add butter and salt if you want to be fancy, etc. The computer could also tell you when to turn, baste, and raise or lower the temperature for a crisp, juicy Henny Penny. When that old machine starts chirping, "Soup's on," you'll have a meal cooked to perfection.

And if an old flame has called you when you should have been thinking of the one under the chicken, throwing off your timing with the latest news for his shrink/her gynecologist, that irrepressible computer of yours will speak up and save the day.

Talk about home computing! This talking computer is the real thing.

If you have children that can speak, a computer with an ear to the kids' rooms may be a nice way to keep them secure and out of your parties. They may think it's fun to tell the computer that they're on their way down for a midnight glass of milk. If Nurse Mitzi, or whatever your electric babysitter is called, warned you, you could get the lampshade off Uncle George's head before the little ones got wise to his tippling.

Once the midnight munchers are back asleep, that fascinating machine could help you keep things lively by playing learning games, an improvement over that last family bash when Uncle George forgot to take the shade off the lamp before he attempted to

HOW TO COMMIT KATAKANA IN FIFTY-ONE SOUNDS

Japan has an interesting approach to writing down the spoken word. The Japanese use two separate systems of writing. One system, Kanji, includes approximately twenty thousand ideographic symbols borrowed from the Chinese. Each character represents an entire word. The other system is based upon the phonetic transcription of words, that is, each symbol or letter represents a sound or syllable. The word *hotel,* for example, is written *ho-te-ru.* This phonetic system of writing utilizes fifty-one symbols which represent the fifty-one sounds of the Japanese language. When used for writing native Japanese words, the system is called Hiragana. But used for transcribing foreign words, it is called Katakana; this effort to transcribe foreign words as closely as the Japanese sound symbols will allow is less than two hundred years old.

Currently, the educated Japanese use both ideographic and phonetic symbols, sometimes in a single written document. Incredibly complex typewriters are equipped with the characters of Kanji, Hiragana, and Katakana. Eventually, however, the phonetic symbols will replace the ideographs. Not only do students balk at learning the twenty thousand ideographic characters, but the Minister of Education has also limited the number of Kanji symbols necessary for daily life (for example, reading the newspaper) to approximately one thousand.

Modern technology will undoubtedly hasten the demise of Kanji. Processing ideographs with computers is a costly procedure. Talking computers, moreover, communicate phonetically for the most part. The phonetic approach to language is practical and the Japanese are a very practical people.

THE GADGETS

Speech Recognizer

SpeechLab
Kit $249 Assembled $299
Heuristics, Inc.
900 North San Antonio Road
Los Altos, CA 94022

Speech Synthesizers

Model 1000
$380
AI Cybernetic Systems
P.O. Box 4691
University Park, NM 88003

CT-1
$395
Computalker Consultants
P.O. Box 1951
Santa Monica, CA 90406

HANDIVOICE
Less than $2,000
Votrax VS-6
Approximately $3,500
Votrax Division of Federal Screw Works
500 Stephenson Highway
Troy, MI 48084

JAPANESE KATAKANA SYLLABARY

		M	N	S	Z	P	B	T	D	K	G	Y	R	H	W
A	ア	マ	ナ	サ	ザ	パ	バ	タ	ダ	カ	ガ	ヤ	ラ	ハ	ワ
I	イ	ミ	ニ	シ	ジ	ピ	ビ	チ	ヂ	キ	ギ	イ	リ	ヒ	ヰ
U	ウ	ム	ヌ	ス	ズ	プ	ブ	ツ	ヅ	ク	グ	ユ	ル	フ	ウ
E	エ	メ	ネ	セ	ゼ	ペ	ベ	テ	デ	ケ	ゲ	エ	レ	ヘ	ヱ
O	オ	モ	ノ	ソ	ゾ	ポ	ボ	ト	ド	コ	ゴ	ヨ	ロ	ホ	ヲ
UN	ン														

get into his muffti and split his lip sucking on a 75-watt softwhite.

If you've got a home workshop and if your computer is equipped to turn on line current, you can hook your system to the lathe. When you say "stop" the lathe stops. If at first you don't trust you computer to turn the lathe on, you could make things safer by insisting on a little dialogue full of "Roger" and "Wilco" stuff.

This is not to say that handy emergency switches shouldn't be in your shop, but your computer-with-an-ear will respond only to one voice—a nice thing to know if you have kids who might attempt to mimic your skill with power machinery.

If Nurse Mitzi, your electric babysitter, warned you, you could get the lampshade off Uncle George's head.

A computer which acts in response to a human voice seems to fulfill our fantasies about household robots. Calculation and accounting and games are fun, all right, but an appliance to run other appliances, now that's the computer we've all dreamed about.

A computer that listens can work with a high-noise-environment mike—the kind you wear on a headset—and a talking machine can power earphones as well as a speaker. What you start as home amusement could well end up as a system in an industrial production line—and it probably will. Computers that can talk and listen will soon be used in aircraft and operating rooms.

In addition to its personal and commercial applications, the voice-oriented computer offers great potential to the handicapped. Computers that can talk and listen will be a boon to those who can't use their hands or who have no hands to use.

Even with a limited vocabulary, a talking computer could provide innumerable services for the blind. A machine that could talk, listen, and emboss braille would facilitate communications both directly and indirectly.

A system that could ask, "Who's there?" and identify the caller by voice would make a superb doorman for the deaf. And imagine a machine that would permit telephone contact for those who can't hear.

Although good systems for the handicapped would take some time to perfect, such machines are in the not-too-distant future. Votrax, for example, recently announced HANDI-VOICE, a portable hand-held terminal with a voice synthesizer that produces phrases and complete sentences from a fixed vocabulary of 1,000 words (plus phonetic potential for more). By means of a throat pick-up, blow switch, palm switch, or head switch, even stroke victims who suffer complete aphasia and those who are totally paralyzed can communicate.

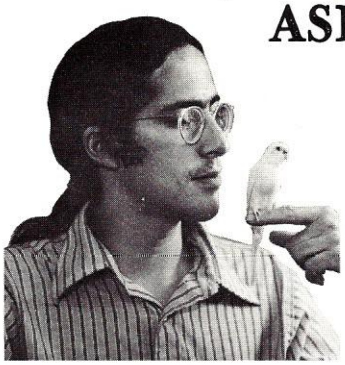
New frontiers have been opened, and they are as accessible to the amateur as they are to the professional. Dedicated hobbyists will undoubtedly make some of the important contributions in computing, not least of all in areas made possible by the advent of speech-processing hardware.

Personally, I'm waiting for an editing machine I can talk to. Even though I've grown to love the typewriter with its body English and pinball-machine noises, when the right gadget comes along I'll run out and buy one. Who knows, I might even build the contraption myself.

What will you talk your computer into doing? ▼

PROMqueries

HAVE A QUESTION? ASK ROM



by Eben Ostby

Dear ROM:

Getting involved with computers, I seem to be unavoidably tripping across terms I'm not familiar with. I'm sure they are self-explanatory to anyone who uses them, but could you tell me exactly what an *op-amp* is? Also what does OEM mean? Is FORTRAN an acronym and if so, what does it stand for?

Brent Hatley
Houston, Texas

Dear Brent,

An op-amp is something you'll frequently find in analog-nondigital electronics. Analog electronics are "traditional" electronics—the type you'll find in your stereo, most likely. An op-amp, or operational amplifier, is a basic amplifier circuit, which can be hooked up in a variety of ways to provide a variety of functions. They have long been produced as integrated circuits, and they are now so cheap—about $100 each, or less—that it's cheaper and easier to use an op-amp with some additional circuitry for many applications than it is to design a separate circuit for the application. So they've become very common.

OEM is an abbreviation for "original equipment manufacturer." OEM equipment, then, is equipment that is produced for inclusion as part of some more elaborate equipment. For instance, an OEM computer module is usually a bare-bones processor that is designed for inclusion as the heart of a computerized piece of equipment. You'll find, for instance, that the OEM version of DEC's PDP-11/03 is the LSI-11, which is a computer sans *case, switches, lights, power supply, etc.*

FORTRAN is indeed an acronym—one of the computer field's first. It stands for FORmula TRANslator—a good name for the early compiler, although compilers do much more than that.

ROM

Dear ROM:

What is a *compiler* and why is it so important in programming?

Dean Rice
Santa Barbara, California

Dear Dean,

A compiler is an elaborate program which has the sole job of translating programs. Since a computer executes not BASIC statements or APL functions (or whatever) directly, but interprets coded numbers as being instructions, any time you write a program in a language like BASIC, it must be translated, or compiled, from English-like statements or mathematical expressions into the computer's own coded instruction set. A compiler lets you write $A = B + C$ *instead of 7600 1020 1021 3017 (or whatever the computer requires). In addition, a compiler lets you name locations in the computer's memory and refer to them by the names you made up, instead of forcing you to remember the addresses of the locations. It's like the difference between remembering that a house is number 1023 Main Street and remembering that it's a purple Victorian with green shutters.*

In effect, a compiler simplifies your code-writing to the point that you can think about the problem at hand, and not about the mechanics of your computer.

ROM

Dear ROM:

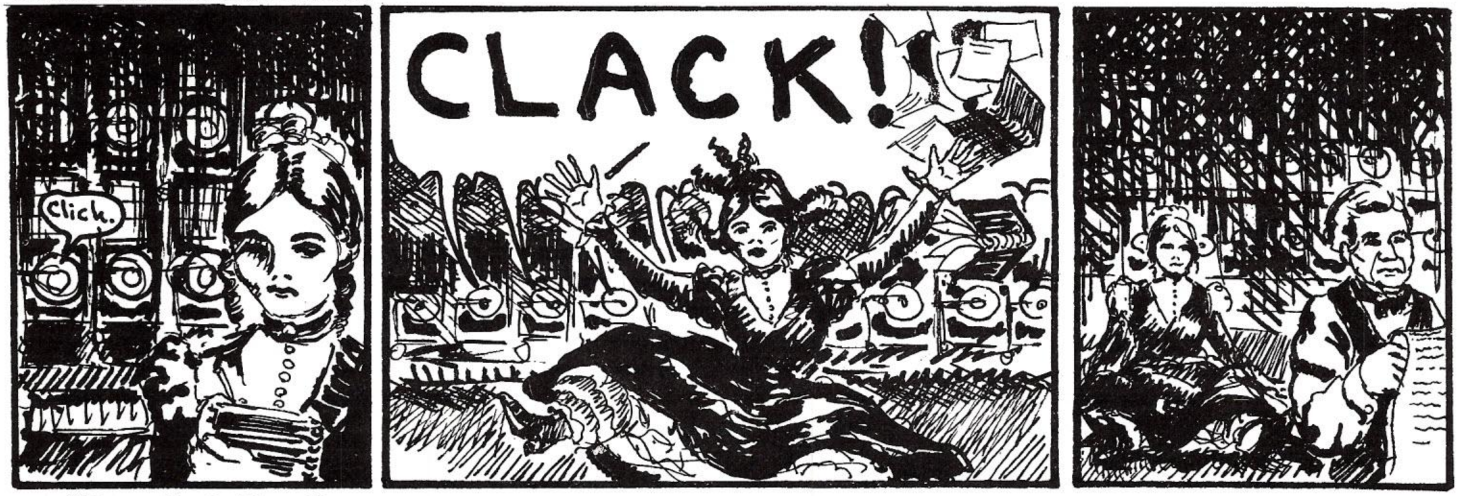
How does an LED work? Also, I know the numerical display on my calculator uses seven segments to make up the different numbers. Is there a segmented display like this for the alphabet? If so, how many fragments does it use?

Steve Driscoll
Boston, Massachusetts

Dear Steve,

The trouble with explaining how an LED works is that it's basically a diode. A diode is a semiconductor device which consists of two regions of a semiconductor. One re-

BABBAGE AND LOVELACE



"Not really, Babbage!"

"Yes, Carry look-ahead."

Cartoon by Steve Gerling, story line by Andrew Singer *gion has been modified, or* doped, *so that it contains a few more electrons than a region of the pure semiconductor would have. The other region is doped in the opposite way: it has a few less electrons than a pure semiconductor would have. When a current is applied to the semiconductor, across the region where the two differently doped regions meet, current flows only when the positive current is applied to the region with fewer than normal electrons. When current is applied in the other direction, no current flows—in fact, there's a sort of running battle inside the diode as to which way the current should flow, so none does. In an LED, the current going across the junction happens to produce light.*

As for LED displays, the only ones we've seen, other than the numeric seven-segment ones, are composed of little dots, roughly forty of them, that are arranged in a five-by-eight matrix. Each little dot is, of course, an LED. By picking out the right combination of dots, it's possible to display nearly any letter or character. A segmented display would be impractical for alphabetics, I suspect.

ROM

Dear ROM:

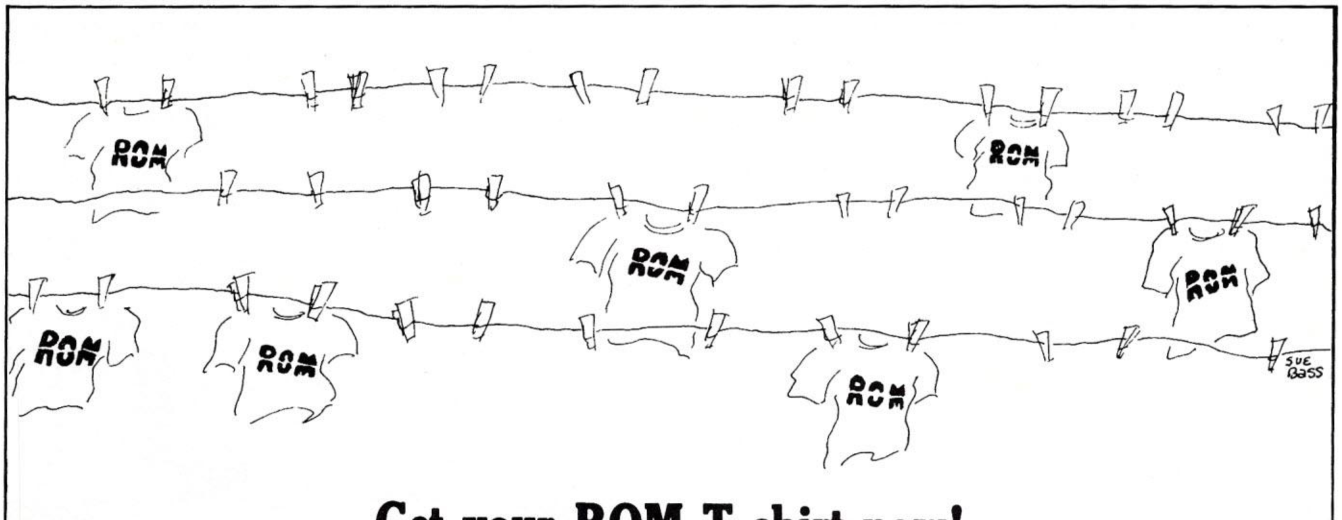
I'm just getting started in computers and one of the things I would like to use mine for, once I get it, is to replace my telephone answering machine. Seems to me I should be able to program the computer so that instead of answering the phone and recording the message, it would dial a new number (say I'm down at the corner bar for the evening), in effect forwarding telephone calls. Would this work? If so, how do I do it? Is it legal? (So little seems to be these days.)

Paul Lorenzo
Wooster, Ohio

Dear Paul,

That's the sort of project I wish I had time for. I think it would be both possible and legal, but its legality might hinge on your using an adaptor for the telephone line that effectively isolates the computer from the telephone. The computer would need to recognize the high-voltage signal that the phone company uses to ring the phone and simulate someone picking up the phone. The phone company recognizes that the phone has been picked up by checking for a sudden drop in the resistance that the phone has—in other words, you'd have to be able to change the apparent resistance of the system automatically. Then you'd have to be able to do the reverse on a second line that you'd use for calling out. If you have Touch-Tone service, your system would have to include an appropriate synthesizer (Newtech Computer Systems' music board is supposed to be able to do this). If your phone has a dial, you'll have to be able to interrupt the line connection at intervals of one tenth of a second for each digit, with a longer space between each set of "clicks" that make up a digit. Your phone company should have information about interfaces to make the whole thing legal—but don't expect them to be too happy if you play around with their lines.

ROM



Get your ROM T-shirt now!

Order from:
ROM Publications Corp.
Route 97
Hampton, CT 06247

Please ship to:
Name__________
Address__________
City__________ **State** __________ **Zip**__________

☐S Qty.__________
☐M Qty.__________
☐L Qty.__________ **☐Child S Qty.**__________
☐XL Qty.__________ **☐Child L Qty.**__________
☐Check or money order enclosed.
☐Master Charge ☐BankAmericard
Exp. date Card#

$5 ppd.
2 for $9 ppd.

Please allow 4 to 6 weeks for delivery.

Legal ROMifications

MICRO AND GOLIATH

by Peter Feilbogen, attorney at law

There is an up-to-date version of the story of David and Goliath which is much more complicated than the original, with the antagonists proportionately smaller than David and certainly much larger than Goliath.

Our legal David and Goliath story begins in 1934 when the Federal Communications Act created the Federal Communications Commission. This Commission, commonly referred to as the FCC, was given authority over "regulating interstate and foreign commerce in communications by wire and radio...." Needless to say, the Bell System was one of the largest and most important subjects of regulation by the FCC.

People, especially Americans, love to invent and improve devices, to modify them, and generally to tinker. This is especially true when there is a large pot of gold available to the successful. The telephone, there in front of everyone's eyes, is certainly a magnet for invention and possible pecuniary gain.

But the FCC and the Bell System effectively squelched anyone who wished to improve the telephone. A tariff, effective through the early 1950s, read as follows:

> "No equipment, apparatus, circuit or device not furnished by the Telephone Company shall be attached to or connected with the facilities furnished by the Telephone Company, whether physically, by induction or otherwise...."

Along came Hush-a-Phone Corporation in the mid-fifties, which produced a simple device that, when attached to the speaker portion of the telephone, would provide the user greater privacy—a third person would be unable to hear the conversation. After great flurry, expense, and time, the FCC's ruling that the new product was "deleterious to the telephone system" was overruled by the Court of Appeals in the District of Columbia. The opinion spoke of "unwarranted interference with the telephone subscriber's right reasonably to use his telephone in ways which are privately beneficial without being publicly detrimental," and this became the new standard.

The Bell System then amended its tariff to read as follows:

> "...Nothing herein shall be construed to permit the use of a device...to interconnect any line or channel of the telephone company with any other communications line or channel of the company or of any other person."

Next came the Carterphone. This was a device which serviced remote radio telephones through a base station. Once again there was substantial flurry, expense, and time expended. The case itself was settled out of Court. The following guidelines came into effect:

1. Restraints imposed upon the customer's right to interconnect were eliminated.
2. Case by case exceptions were eliminated.
3. It was established that customers could not interfere with telephone company service or be dangerous or harmful to it.
4. It was decided that network control signals, e.g. dial pulsing, must be performed by telephone company equipment.
5. From now on, connections must be made by telephone company equipment unless they are made acoustically or inductively.
6. Minimum network protection criteria were established.

In effect the new guidelines fostered the manufacturing of equipment by non-Bell companies.

In 1966 the FCC, as a result of its report, decided that it would not regulate the data-processing industry where the "primary thrust" is data processing and not message switching. In doing so the FCC set forth a definition of data processing, message switching, hybrid data processing service, and hybrid commerce service. The overall effect was once again to increase competition.

Since then, there has been a continual erosion of the Bell System's control over devices which are appended to the subscriber's equipment. Recently, the American Telephone and Telegraph Co. issued a memo in which the Bell System, in effect, has accepted the actual and the inevitable which has been building since the fifties.

To a computer enthusiast, this means a great deal more than having the ability to put privately-owned extension phones in your home. It will mean a whole revolution in peripheral equipment, which will not only be available for interfacing with the telephone to facilitate computer-to-computer data exchange—now it will be legal. ▼

Solution to last month's PROMpuzzle

L	I	S	P	■	B	A	S	■	P	A	M	■	S	A	T	E
A	S	E	A	■	U	S	A	■	A	G	E	■	H	I	E	R
W	I	T	S	■	S	I	L	I	C	O	N	■	O	D	D	S
S	T	A	T	E	■	■	E	N	T	■	■	E	R	A	S	E
■	■	■	E	A	R	O	M	S	■	D	O	L	T	■	■	■
C	A	B	■	T	A	B	■	■	■	I	R	S	■	L	O	T
A	L	I	■	E	D	I	T	S	■	O	L	E	■	E	L	I
P	A	P	E	R	■	■	H	A	N	D	Y	■	P	A	D	S
■	■	O	B	S	■	L	E	V	E	E	■	A	R	K	■	■
B	U	L	B	■	T	O	N	E	R	■	■	N	E	A	R	S
I	F	A	■	L	A	S	■	S	O	U	S	A	■	G	O	O
T	O	R	■	U	P	S	■	■	■	S	O	L	■	E	M	S
■	■	■	P	R	E	Y	■	R	A	N	D	O	M	■	■	■
P	H	A	S	E	■	■	C	A	L	■	■	G	A	T	E	S
R	O	B	E	■	C	H	A	R	G	E	S	■	C	O	R	E
A	L	E	C	■	A	I	R	■	O	R	A	■	R	A	N	T
M	E	T	S	■	M	E	D	■	L	A	G	■	O	D	E	S

futuROMa

THE NEUTRINO CONNECTION

by Bill Etra

Satellites have put considerably more bounce to the ounce in today's information transmission. With microwave beams being deflected accurately off satellites and caught by targeted microwave dish antennas at the receiving location, both the quantity and distance of mass communication have taken a quantum leap forward. However there are limitations.

The first is that the most efficient type of satellite is an anchored satellite, one which stays in a relatively fixed location over the same spot on the earth. As a result, possible transceiver locations are limited to an area within direct sighting lines of the satellite. The transmission distance is also severely limited because microwaves travel only in straight lines and cannot bend around the earth's curvature. In order to get a message around the earth, you have to relay it. The transmission is bounced from one satellite to the next, necessitating a chain of satellites that sit like a string of pearls above various points on the earth. Still, satellite transmitters typically have a bandwidth of up to better than four megahertz which means a lot of information can be handled at one time. A similar video picture in a four-megahertz bandwidth, for instance, will allow you to send roughly 500,000 bytes per sixtieth of a second.

Another negative factor is money. Satellite communications are very costly. From the initial launch to routine maintenance, it causes a continuous and nasty dent in the cash flow. For example, one reason for building the rather expensive space shuttle is to help maintain our satellite system, which is becoming more and more vital for communication.

Finally, there's the pollution question. Microwaves are a form of pollution. They are dangerous. At the moment, we blissfully ignore this subject but, basically, anyone in the direct path of a strong, narrow microwave link is getting fried to one degree or another. I still remember visiting a microwave reception station for Teleprompter and Cable TV on Cape May where the engineers used to remove a section of the microwave guide to roast their hot dogs and to warm their sandwiches on for lunch. The thought that in the not very distant future every house could have a small microwave dish antenna on its roof for direct transmission and reception of information is alarming. Both heat pollution and microwave pollution will become a problem when we start doing things like this.

A much more economical, if currently overlooked, system for data transmission involves utilization of TV transmission time which is currently dissipated. A whole network is out there ready to be used, for almost nothing, if you take into account one factor—the video blanking pulse.

The video blanking pulse in the standard TV operation is roughly ten microseconds every horizontal video line. For a brief period of time, the beam that writes the line on your picture tube passes into an area which you don't see. When it finishes writing half the lines in the picture, the beam moves to the bottom and shuts itself off for approximately fourteen hundred microseconds while it returns to the top and starts writing lines again, horizontal lines. Now, this means that every sixtieth of a second there are fourteen hundred microseconds during which you could comfortably transmit data over regular video transmitters. It has been proposed that information links from city to city be carried on by the television networks during these blanking times. The networks are already using the blanking time to transmit all the new signals required by automatic alignment sets and to send time data for certain TV sets that receive it and other information. So it clearly works. But currently it is vastly underutilized. Whole pages of data can be transmitted in this blanking time, several pages every sixtieth of a second.

The concept has been tested in several places, including New York State, where the Public Broadcasting System was used. It is a very real way of making more computer data available for the personal system on a worldwide, or at least in this case U.S.-wide basis.

Another method that's coming into its own in data transmission is cabling. Fiber-optic light cabling with laser transmission is being used in a lot of places. It has a much greater bandwidth than convention cable, although it has the same built-in problem. Somebody has to lay the cable. Also laser links, like microwaves, have a polluting effect. There is a limit to the amount of stray laser light you want in the atmosphere, just as there is a limit to how much free microwave pollution you'd like.

So the problem boils down to this: although we have assumed that laser and microwave communication is good and safe, it is, in reality, good but not very safe. Piggybacking on video transmission, probably the most practical form of data transmission proposed so far, is not satisfactory since it's not a two-way system. What we need is a new system of high data rate transmission that will bring costs down to everyman's computer.

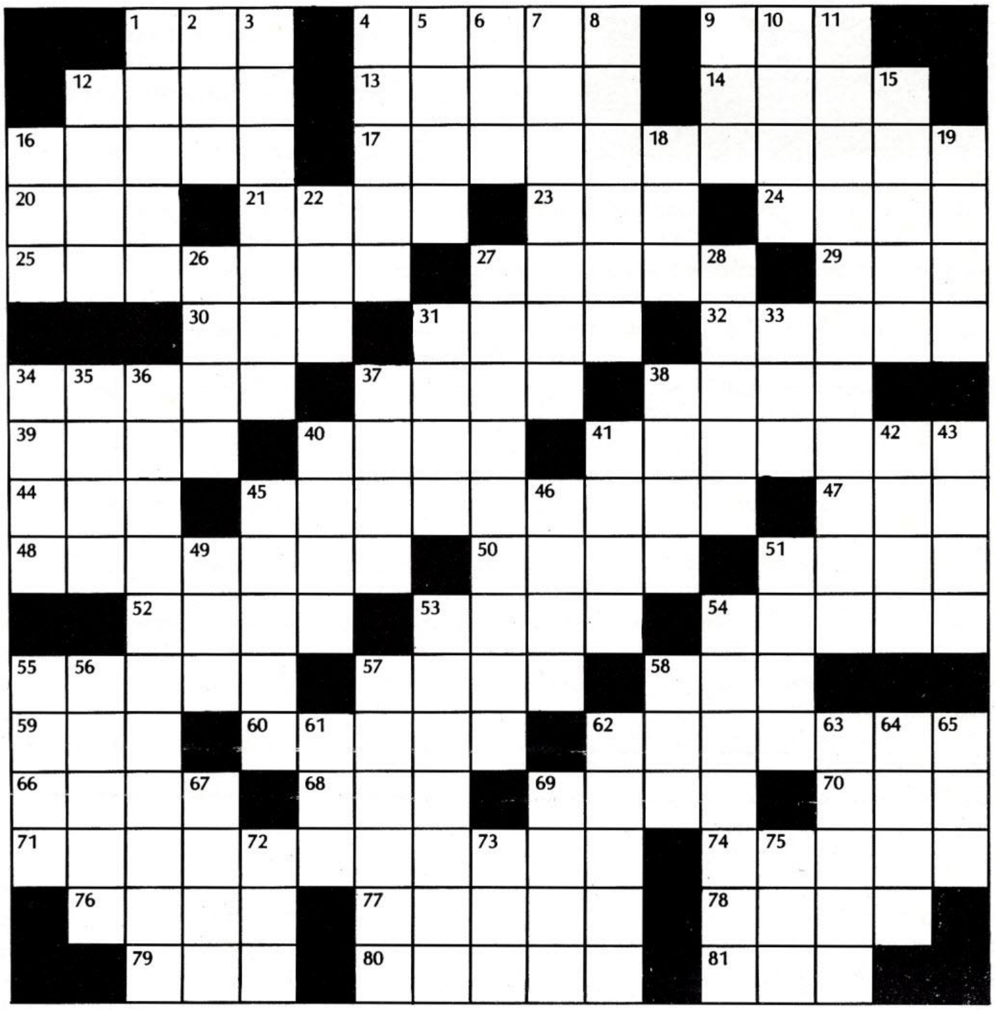
One such system currently being tested by the Navy involves neutrinos. This year an experiment on neutrino detection between Chicago and Washington State is being conducted by physicists Peter Kotzer and James Albers of Washington State University. They expect to detect neutrinos, from the Fermi Accelerator in Chicago, landing in Puget Sound.

Neutrinos pass through the earth's atmosphere, through the earth's oceans, through just about everything. Since they are subatomic particles, there is a reasonable chance of detecting neutrino streams in a liquid medium by looking for the characteristic light flashes that neutrinos cause when entering water. This can be a computer-monitored operation.

At the moment, an enormously large particle accelerator is needed for the development of neutrino beams that are strong enough to be detected. Yet, here is potential for a whole new form of worldwide data transmission, without satellites, microwaves, or lasers. Still a nagging doubt slips slowly over the horizon with the possibility of this form of data transmission as well. How dangerous might neutrinos be to your health? ▼

PROMpuzzle

by Daniel Alber



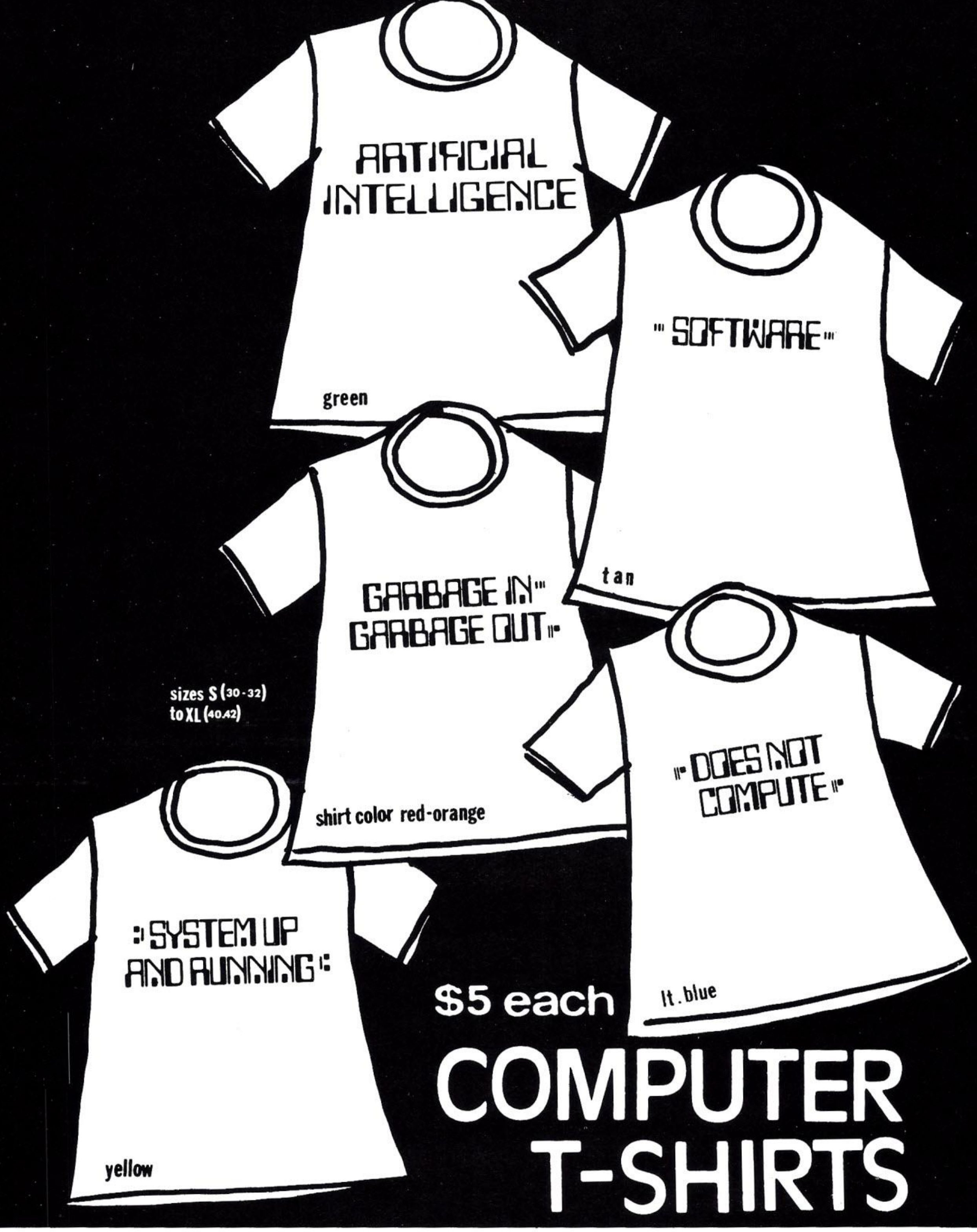
The solution to this PROMpuzzle will appear in next month's *ROM*.

ACROSS

1. Memory-address register (abbr.)
4. Spanish poet
9. Poetic contraction
12. 4096 consecutive bytes
13. Composer Dvorak
14. Cut ____ (2 wds.)
16. Where Seoul is
17. ____ load module
20. Marker light indicator (abbr.)
21. Mines
23. Cathode ray tube (abbr.)
24. To examine point-by-point in logical sequence
25. Manual computer control portion
27. Belonging to the Institute of Electrical and Electronics Engineers
29. ____even check
30. Color
31. Electrosensitive programmings
32. J.D. Salinger character and others
34. Group of eight
37. Highlander
38. Piece of data
39. Sector
40. Stringed instrument
41. Devices making output equal to input
44. Epoch
45. Imitates a computer system
47. Teachers' group (abbr.)
48. Type of transducers
50. War god
51. Remote data transmitters
52. Food fish
53. Explosives
54. Circuit saftey devices
55. Type of circuit
57. Eastern rulers (var.)
58. Hawaiian dish
59. Electronic display unit
60. Very thin
62. ____ delay time
66. Automatic output controls (abbr.)
68. "____ Got a Secret"
69. Slangy negative
70. Caviar
71. Minicomputer addressing mode
74. Self running computer
76. Fashion name
77. Hard work
78. Mine entrance
79. Stitch
80. Rock
81. Mesh

DOWN

1. Nautical (Fr.)
2. Period in history or geology
3. Type of computer display
4. Gigantic
5. Singles
6. Highway (abbr.)
7. Notion
8. Dens (Fr.)
9. Nicotine's partner
10. Gershwin and others
11. Type of computer request
12. Sport
15. Wooded area
16. Kilomegacycle (abbr.)
18. Summer in Paris
19. Goals
22. ____ de France
26. Lean-to
27. ____ oxide-isolation
28. Bristles
31. Pale color
33. Choice (abbr.)
34. Has bills
35. Have feelings
36. Electrical flow device
37. Our uncle et al.
38. Frosts
40. Employ
41. Holy women (abbr.)
42. Network
43. Back talk
45. Lone flights
46. ____ and crafts
49. Oneself in Marseilles
51. Destroy
53. Theatre sign (2 wds.)
54. A computer programming language
55. Harvest
56. Love
57. Cuts off
58. Play on words
61. Concealed
62. Jungle cat (Fr.)
63. Planet path
64. Twelve inches
65. Field effect transistor (abbr.)
67. Locale
69. Soon
72. Recent
73. ____ to the hills____ (abbr.)
75. Poem


ARTIFICIAL INTELLIGENCE
green
"SOFTWARE"
tan
"GARBAGE IN"
GARBAGE OUT"
shirt color red-orange
"DOES NOT COMPUTE"
lt. blue
:SYSTEM UP AND RUNNING:
yellow

sizes S (30-32)
to XL (40,42)

$5 each

COMPUTER T-SHIRTS

name ____________________

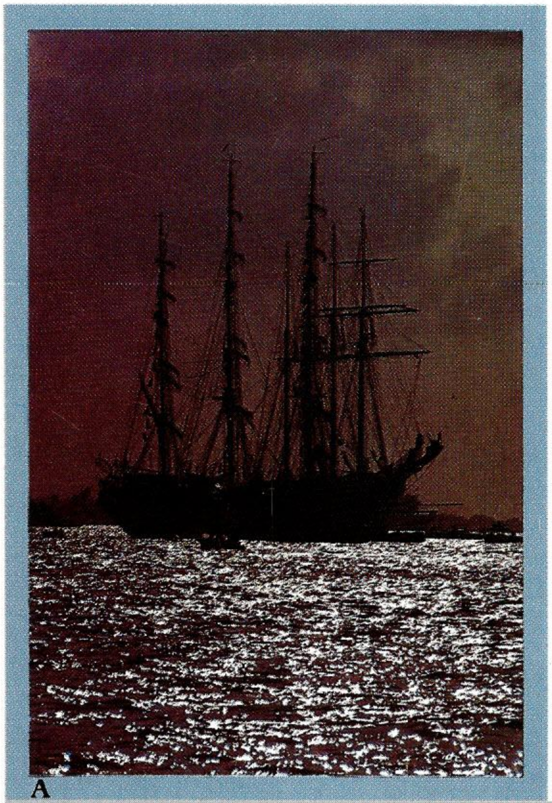
address ____________________

city ________ state ____ zip ______

mail to MARTHA HERMAN 114 W. 17 STREET
NEW YORK 10011
(212) 691-2821

quantity	chest size	design	$	
add .60 per shirt for shipping				
total enclosed				

A magnificent first edition . . . pictorial memories of a moment America will never forget . . . or see again

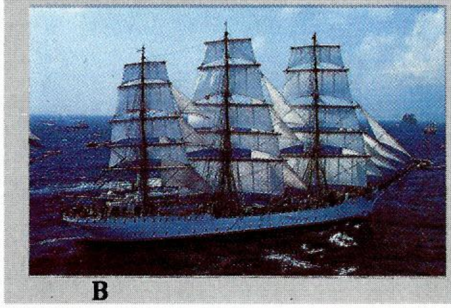


A

THE SAILING SHIPS

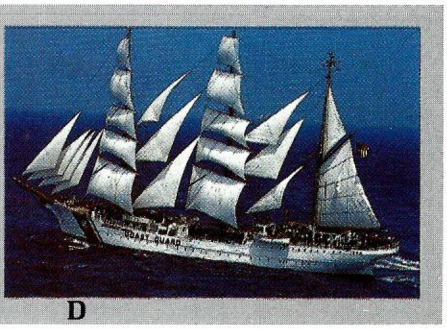
Professional quality lithographs of the majestic Bicentennial Ships, now available to the public in an exclusive, limited edition from America's foremost engraver-printer, COLLIER GRAPHICS.

handsomely matted in silvery metal frames; $25 each **unframed, $10 each**



B

C



D

THIS IS A LIMITED EDITION . . . ORDER TODAY WHILE THEY LAST

Never before offered to the public, these magnificent 12x19 full color lithographs of the tall-masted SAILING SHIPS were originally photographed for professional use only!

Now, you can own and enjoy their historic beauty, their incredible clarity, color and reproduction, which gives them almost a three-dimensional quality. And they are only available from COLLIER GRAPHICS — who provide the superb color graphics for America's leading advertising agencies and publishers.

Perfect for your home, boat or office. A lasting gift. Order a set for yourself and one for your children . . . to keep always.

COLLIER GRAPHICS INC., 240 West 40th Street, New York, New York 10018

Please rush the following SAILING SHIPS, securely packaged and postage prepaid, with the understanding that my purchase is UNCONDITIONALLY GUARANTEED by you if the order is returned within 15 days of delivery:

Send me the following print(s). Quantity ________ Price ________ Total ________

A. Christian Radich ________

B. Danmark ________

C. Kruzenshtern ________

D. Eagle ________

Please add $2.50 for shipping and handling for framed print(s) or $1.00 for unframed print(s).
(N.Y. State residents add applicable tax, NYC residents add 8%. Allow 6 weeks for delivery.)

Name ________

Address ________

City ________ State ________ Zip ________

Enclosed, check or money order for $________
Or charge my credit card ________ Master Charge ________ BankAmericard

Credit Card # ________ Inter Bank # ________ Expiration Date ________

Signature X ________